remote sensing

MDPI

Article

Spatial-Adaptive Siamese Residual Network for Multi-/Hyperspectral Classification

Zhi He [1,*] and Dan He [2]

[1] Guangdong Provincial Key Laboratory of Urbanization and Geo-Simulation, Center of Integrated Geographic Information Analysis, Southern Marine Science and Engineering Guangdong Laboratory (Zhuhai), School of Geography and Planning, Sun Yat-sen University, Guangzhou 510275, China
[2] City College of Dongguan University of Technology, Dongguan 511700, China; hed@ccdgut.edu.cn
* Correspondence: hezh8@mail.sysu.edu.cn; Tel.: +86-020-8411-3057

Received: 12 April 2020; Accepted: 16 May 2020; Published: 20 May 2020

Abstract: Deep learning methods have been successfully applied for multispectral and hyperspectral images classification due to their ability to extract hierarchical abstract features. However, the performance of these methods relies heavily on large-scale training samples. In this paper, we propose a three-dimensional spatial-adaptive Siamese residual network (3D-SaSiResNet) that requires fewer samples and still enhances the performance. The proposed method consists of two main steps: construction of 3D spatial-adaptive patches and Siamese residual network for multiband images classification. In the first step, the spectral dimension of the original multiband images is reduced by a stacked autoencoder and superpixels of each band are obtained by the simple linear iterative clustering (SLIC) method. Superpixels of the original multiband image can be finally generated by majority voting. Subsequently, the 3D spatial-adaptive patch of each pixel is extracted from the original multiband image by reference to the previously generated superpixels. In the second step, a Siamese network composed of two 3D residual networks is designed to extract discriminative features for classification and we train the 3D-SaSiResNet by pairwise inputting the training samples into the networks. The testing samples are then fed into the trained 3D-SaSiResNet and the learned features of the testing samples are classified by the nearest neighbor classifier. Experimental results on three multiband image datasets show the feasibility of the proposed method in enhancing classification performance even with limited training samples.

Keywords: remote sensing; classification; stacked autoencoder; superpixel; Siamese network

1. Introduction

Remote sensing [1–4] has played an important role in Earth observation problems. The ever-growing number of multispectral and hyperspectral images acquired by satellite sensors facilitates a deeper understanding of the Earth's environment and human activities. Rapid advances in remote sensing technology and computing power have contributed to the application of multiband images in environmental monitoring [5], land-cover mapping [6,7], and anomaly detection [8]. Classification, which labels each pixel with a specific land-cover type, is one of the fundamental tasks in remote sensing analysis [9–11]. A large number of labeled training data are usually required to achieve satisfactory classification performance. However, it is time-consuming and expensive to label the remote sensing samples in practice.

In the remote sensing community, researchers have proposed various supervised approaches to classify the multiband images. Existing methods can be roughly divided into two categories. The first category is spectral classification methods, which classify each pixel independently by utilizing the spectral features. State-of-the-art methods include the support vector machine (SVM) [12]

and its variations [13,14]. Although the spectral methods have proven to be useful for multiband images classification, they fail to fully explore the spatial information, and therefore result in salt and pepper-like errors. The second category is spectral-spatial classification methods, which use both spectral and spatial information for more accurate classification performance. In this category, additional spatial features extracted by diverse strategies are integrated into the pixel-wise classifiers. Widely-used spatial extraction methods include the morphological profiles [15], attribute profiles [16], gray-level cooccurrence matrix [17], Gabor transform [18], and wavelet transform [19]. Image segmentation can also be applied to obtain the spatial neighbors. For instance, superpixel segmentation groups pixels of multiband images into perceptually meaningful atomic regions which are more flexible than the rigid square patches [20,21]. Many classifiers are also proposed for spectral-spatial classification, such as the joint sparsity model (JSM) [22], multiple kernel learning (MKL) [23], and Markov random field (MRF) modeling [24]. Moreover, note that the remote sensing images can be modeled as three-dimensional (3D) cubes with one spectral dimension and two spatial dimensions. Researchers have developed some spectral-spatial methods under the umbrella of tensor theory, such as the tensor discriminative locality alignment (TDLA) method [25], tensor block-sparsity based representation [26], and local tensor discriminative analysis technique (LTDA) [27]. Spectral-spatial methods have proved to be more effective than the spectral methods as they can exploit the object shape, size, and textural features from the multiband images.

However, most of the above-mentioned methods cannot classify the multiband images in a "deep" fashion. In recent years, deep learning [28–31] has attracted extensive attention with the rapid development of artificial intelligence. Deep learning can hierarchically learn the high-level features, and therefore has achieved tremendous success in remote sensing classification. Typical deep architectures include the stacked autoencoder (SAE) [32], deep belief network (DBN) [33], residual network (ResNet) [34] ,and generative adversarial network (GAN) [35]. Among all deep learning-based methods being developed over the past few years, convolutional neural networks (CNNs) have become a preferable model in the current trend of multiband images analysis due to the fact that convolutional filters are powerful tools to automatically extract the spectral-spatial features. For instance, a one-dimensional CNN (1D-CNN) method was proposed in [36] for hyperspectral classification in the spectral domain. A spectral-spatial classification method was proposed in [37] by combining the principle component analysis (PCA), two-dimensional CNN (2D-CNN), and logistic regression. [38] improved the performance of remote sensing image scene classification by imposing a metric learning regularization term on the CNN features. The 3D-CNN was applied in [39] to learn the local signal changes in spectral-spatial dimension of the 3D cubes, while [40] explored the performance of deep learning architectures for remote sensing classification and introduced a 3D deep architecture by applying 3D convolution operations instead of using 1D convolution operations. In [41], residual blocks connected every 3D convolutional layer by identity mapping to gain improved classification performance. Moreover, the Siamese networks [42] composed of 2D-CNN/3D-CNN or 2D-ResNet/3D-ResNet [43–46] were also proposed for hyperspectral classification. The Siamese architecture differs from other deep learning-based methods in that it is directly trained to separate the similar pairs from the dissimilar in feature space. By simultaneously learning the similarity metrics, the Siamese networks are effective for distinguishing different land-covers in remote sensing data. However, all those Siamese networks performed pixel-based classification of hyperspectral data by choosing the neighborhoods of a sample as the original features. Although CNN and its variations have been successfully employed in classification of multiband images, training, or testing samples are usually fixed at the center of the neighborhood windows and the neighboring samples in each patch are assumed to be consisted of similar materials. However, in heterogeneous regions, especially around the boundaries, neighboring samples often belong to different classes, which go against the basic assumption of the deep learning-based methods.

In this paper, we propose a 3D spatial-adaptive Siamese residual network (3D-SaSiResNet) for classification of multiband images. The main steps of the proposed method are twofold,

i.e., construction of 3D spatial-adaptive patches and Siamese residual network for multiband images classification. In the first step, we segment the multiband images into spatial-adaptive patches instead of choosing the patches centered on training/testing samples. In the second step, we design a 3D-SaSiResNet to extract high-level features for classification. The 3D-SaSiResNet is composed of two 3D ResNet channels with shared weights and the training samples are utilized to train the network in pairs. The testing samples are then fed into the trained Siamese network to obtain the learned features, which can be classified by the nearest neighbor classifier. The two steps are closely connected, i.e., in the first step, we generate 3D spatial-adaptive patches, which can be subsequently classified by the 3D-SaSiResNet built in the second step.

The main contributions of this work are listed below:

- We obtain 3D spatial-adaptive patches of the multiband images by band reduction and superpixel segmentation. Instead of fixing the samples at the center of the patches, the 3D spatial-adaptive patches provide a more flexible way to fully consider the heterogeneous regions of the remote sensing data, making the neighbors within the same 3D patch more probable to have the same class label.
- We propose a 3D-SaSiResNet method for classification of multiband images. The Siamese architecture of the network combined with 3D ResNet helps to extract discriminative spectral-spatial features from the remote sensing data cube and provides promising classification performance even with limited training samples.

The rest of this paper is organized as follows. In Section 2, detailed descriptions of the proposed method are provided. Section 3 reports experimental results on three multiband image datasets and conclusions are drawn in Section 4.

2. Proposed Method

In this section, we provide the detailed procedure of the proposed method, which contains two main parts: construction of 3D spatial-adaptive patches and 3D-SaSiResNet for multiband image classification (see Figure 1). Both parts of the proposed method are important for multiband image classification. First, compared to fixing the training or testing samples at the center of the patches, constructing spatial-adaptive patches can fully follow the spatial characteristics of the multiband image, making the pixels within the same patch more likely to be in the same category. It is notable that the hyperspectral image usually has more than one hundred contiguous bands, and band reduction is also essential to reduce the computational cost in the first steps. Second, the 3D-SaSiResNet facilitates the extraction of discriminant features and classification of multiband images.

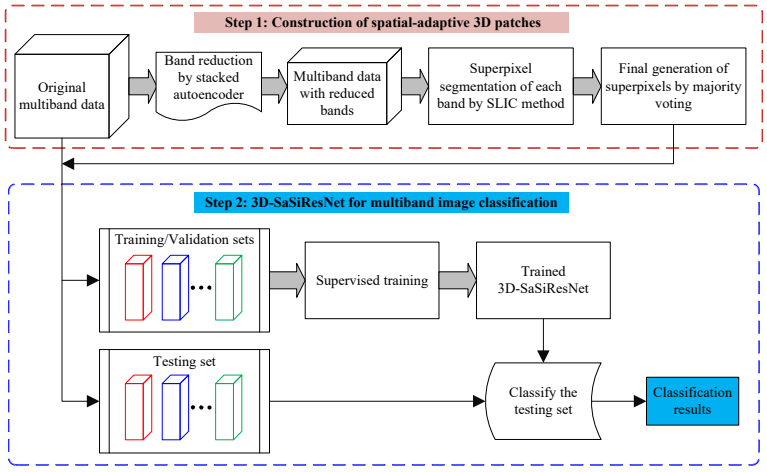


Figure 1. Schematic illustration of the proposed 3D-SaSiResNet.

2.1. Construction of 3D Spatial-Adaptive Patches

In the proposed method, we segment the multiband images into 3D spatial-adaptive patches, ensuring that the neighbors within the same patch are more likely to consist of the same materials. To reduce the computational cost, band reduction implemented by SAE [31,47,48] is first performed on the original images. The reason for utilizing the SAE for band reduction is that the accuracy of superpixel segmentation results is important for the proposed method and the SAE can provide an effective nonlinear transformation and nonlinear band reduction. Letting $\boldsymbol{I}_0 \in \mathbb{R}^{m_0 \times n_0 \times b_0}$ be the original multiband images data, where m_0, n_0, and b_0 represent the number of rows, columns, and spectral bands, respectively, the intrinsic dimensionality of $\boldsymbol{I}_0$ can be estimated by the "*intrinsic_dim*" function in the "*drtoolbox*" [49], which is a public MATLAB toolbox for dimensionality reduction. Supposing the intrinsic dimensionality of $\boldsymbol{I}_0$ is b, we design a SAE network to reduce the band of $\boldsymbol{I}_0$ from b_0 to b. The structure of the SAE used in this paper is depicted in Figure 2, where the input layer $\tilde{\boldsymbol{I}}_0 \in \mathbb{R}^{b_0}$ is the spectral signature of $\boldsymbol{I}_0$. As shown in Figure 2, SAE is a stacked structure of multiple autoencoders and each autoencoder is a single hidden layer feedforward network, whose output aims to reconstruct the input signal. In every single autoencoder, the hidden layer units are less in number than the input and output layer units. As such, the autoencoder network learns a compressed representation of the high-dimensional remote sensing data. Without loss of generality, supposing the input signal of a single autoencoder is $\boldsymbol{X} \in \mathbb{R}^{b_x}$, which is reduced to b_y features representing high abstraction in the hidden layer, and then the $\boldsymbol{X}$ is reconstructed into $\boldsymbol{Z} \in \mathbb{R}^{b_x}$. The above-mentioned process can be formulated as

$$\boldsymbol{Y} = f_y(\boldsymbol{w}_y \boldsymbol{X} + \boldsymbol{\beta}_y) \tag{1}$$

$$\boldsymbol{Z} = f_z(\boldsymbol{w}_z \boldsymbol{Y} + \boldsymbol{\beta}_z) \tag{2}$$

where $\boldsymbol{Y} \in \mathbb{R}^{b_y}$ refers to the compressed data in the encoding step, $\boldsymbol{Z}$ represents the reconstructed data in the decoding step, $\boldsymbol{w}_y$ and $\boldsymbol{w}_z$ denote the weights of input-to-hidden and hidden-to-output, respectively, $\boldsymbol{\beta}_y$ and $\boldsymbol{\beta}_z$ are the bias of the hidden and output units, respectively, and $f_y(\cdot)$ and $f_z(\cdot)$ denote the activation functions.

The autoencoder can be trained by minimizing the following reconstruction error:

$$\min_{\boldsymbol{w}_y, \boldsymbol{w}_z, \boldsymbol{\beta}_y, \boldsymbol{\beta}_z} \|\boldsymbol{X} - \boldsymbol{Z}\|_2^2 \tag{3}$$

The SAE used in this paper is a stacked series of autoencoders layer by layer, where the hidden layer output of the previous autoencoder is taken as the input of the next layer. The parameters of the SAE can be obtained by training each autoencoder hierarchically with the back propagation algorithm in an unsupervised fashion. Moreover, it is worthwhile to note that we use five hidden layers (the number of units is set to 100, 50, 25, 10, and b, respectively) for band reduction of hyperspectral image since the hyperspectral image usually contains hundreds of contiguous spectral bands, too many layers will increase the computing burden, while fewer layers will reduce the extraction effect. As to the multispectral image, only one hidden layer (the number of units is set to the intrinsic dimensionality) is needed since the spectral bands of multispectral image are usually less than 10, and, therefore, it is not necessary to adopt as many hidden layers as the hyperspectral image has.

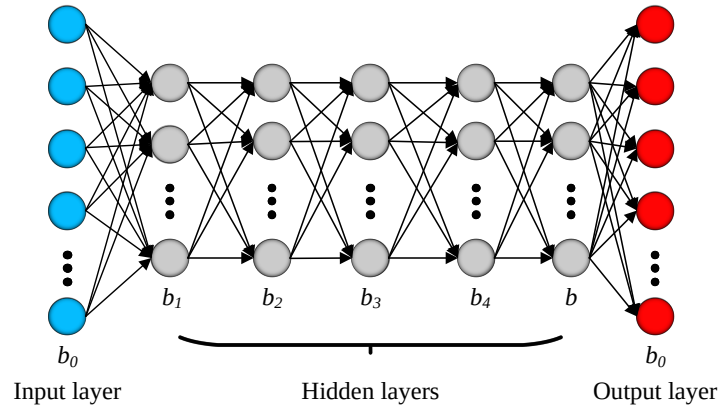


Figure 2. Structure of the SAE (with five hidden layers) for band reduction.

Simple linear iterative clustering (SLIC) [20] is then performed on each of the reduced bands to obtain over-segmented maps. Letting $\boldsymbol{I} \in \mathbb{R}^{m_0 \times n_0 \times b}$ be the multiband image with reduced bands and $\boldsymbol{I}_s \in \mathbb{R}^{m_0 \times n_0 \times b}$ be the superpixel segmentation results obtained by performing SLIC to each of the b band, the final superpixel map $\boldsymbol{S}$ can be generated by Algorithm 1. Moreover, Figure 3 gives an example about how to obtain $\boldsymbol{S}$. There are three bands in $\boldsymbol{I}$, and $\boldsymbol{I}_s \in \mathbb{R}^{7 \times 7 \times 3}$ denotes the segmentation results of SLIC. In case the intersection of all the over-segmented regions where the pixel (p, q) located is not empty, the pixels located at that intersection will be given the same label. Otherwise, a single label is given to the (p, q)th pixel. The above-mentioned procedure is repeated with (p, q) changes from $(1, 1)$ to $(7, 7)$, and the superpixel map $\boldsymbol{S}$ is finally generated by traversing all of the pixels in $\boldsymbol{I}_s$. Moreover, it is important to note that the labels in $\boldsymbol{S}$ denote different segmentation blocks rather than represent the class labels of land covers.

Given the superpixel map $\boldsymbol{S}$, instead of fixing a certain sample $\boldsymbol{s}_0$ at the center, the 3D spatial-adaptive patch of $\boldsymbol{s}_0$ is determined by cropping the minimum bounding rectangle of the superpixel in which $\boldsymbol{s}_0$ lies and extracting the pixels from the input multiband image according to the cropped minimum bounding rectangle. To ensure all the pixels in the same patch have the same land-cover class, those pixels, which are located in the minimum bounding rectangle but do not belong to the same superpixel, are substituted by the mean of the rest pixels in the minimum bounding rectangle. All of the 3D patches are resized to the same window size by bicubic interpolation for the convenience of model training. Specifically, Figure 4 depicts a schematic example of the patch generation procedure. The window size of the patch is set to 3×3. Figure 4a denotes the procedure of extracting a 3D patch by fixing a certain sample $\boldsymbol{s}_0$ at the center, the materials, and land-covers of the neighbors may be quite different from $\boldsymbol{s}_0$, and the number of patches is equal to the number of samples. Figure 4b denotes the procedure of extracting a 3D spatial-adaptive patch, in which the neighbors are more likely to have the same labels as $\boldsymbol{s}_0$. Moreover, all the pixels that are contained within the same superpixel result in the same 3D spatial-adaptive patch, and, therefore, the number of generated 3D spatial-adaptive patches is equal to the number of superpixels. Taking each 3D spatial-adaptive patch as an object in multiband classification, the 3D-SaSiResNet is, in essence, an object-based method rather than a pixel-based method.

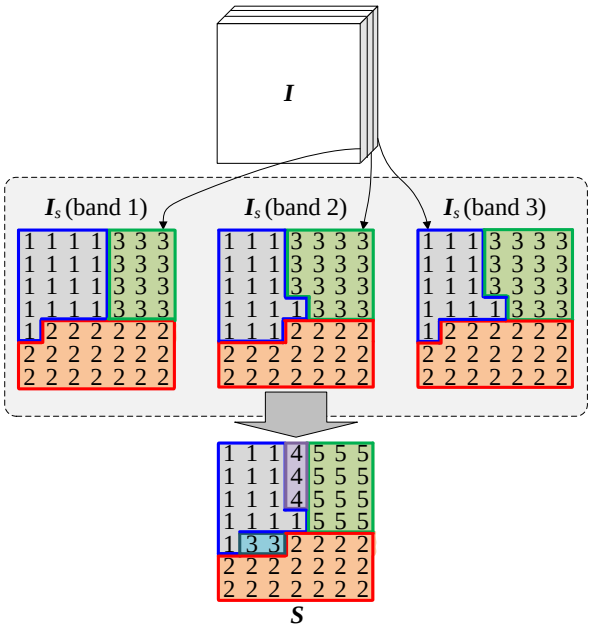
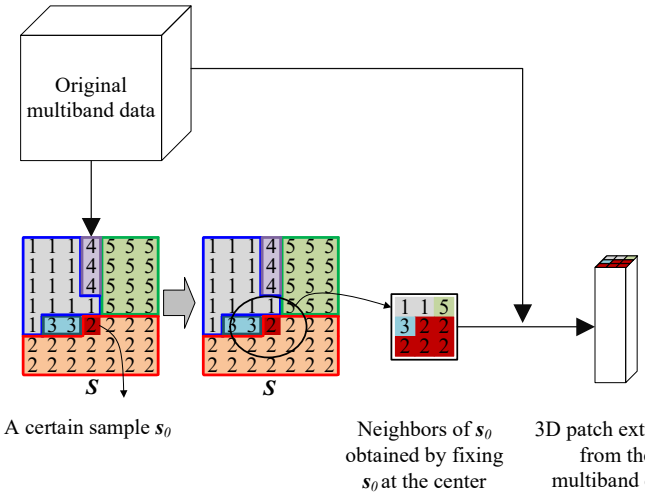


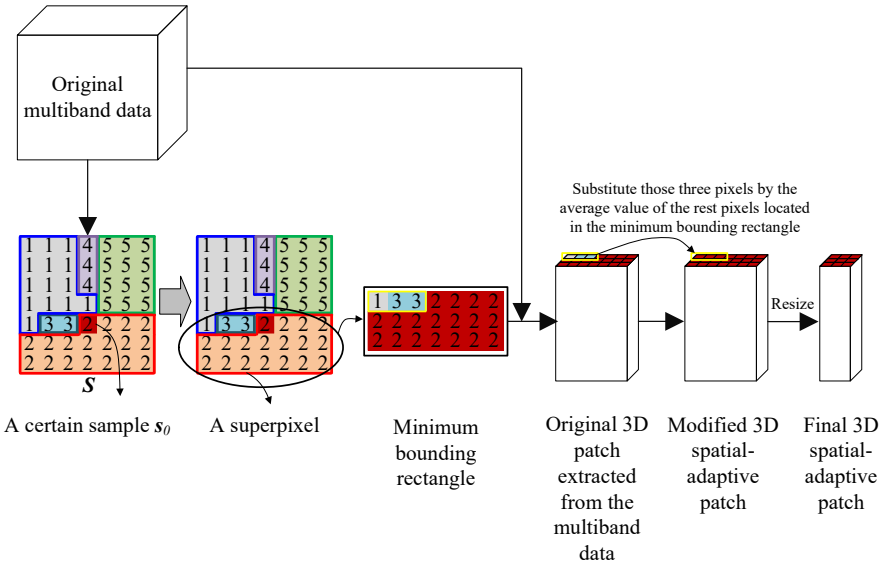
Figure 3. Schematic example about how to obtain the superpixel map S.

Algorithm 1: **Generating the superpixel map** S.

Input: The multiband image I with reduced bands;
Output: The final superpixel map S;
// S is initialized to a $m_0 \times n_0$ matrix whose elements are 0;
Initialize: k=1; $S = \text{zeros}(m_0, n_0)$;
for *i=1 to b* **do**
  // Perform SLIC on the ith band of I;
  $I_s(:,:,i) = \text{SLIC}(I(:,:,i))$;
  // $I_s(:,:,i)$ and $I(:,:,i)$ denote the ith band of I_s and I, respectively;
end
for *p=1 to* m_0 **do**
  for *q=1 to* n_0 **do**
    if $S(p,q)$ *is equal to 0* **then**
      for *i=1 to b* **do**
        Find the coordinates (p_i, q_i) where $I_s(:,:,i)$ is equal to $I_s(p,q,i)$;
        // $I_s(p,q,i)$ denotes the element of I_s located at the pth row, qth column and ith band;
      end
      Calculate the intersection $p_\cap$ of all the $p_i, i = 1, 2, \ldots, b$;
      Calculate the intersection $q_\cap$ of all the $q_i, i = 1, 2, \ldots, b$;
      if $p_\cap$ *and* $q_\cap$ *are empty* **then**
        $S(p,q) = k$;
      else
        $S(p_\cap, q_\cap) = k$;
      end
      $k \longleftarrow k + 1$;
    end
  end
end



(**a**) Extracting the 3D patch by fixing s_0 at the center



(**b**) Extracting the 3D spatial-adaptive patch.

Figure 4. Schematic example of the patch generation procedure. (**a**) extracting the 3D patch by fixing s_0 at the center; (**b**) extracting the 3D spatial-adaptive patch.

2.2. 3D-SaSiResNet for Multiband Image Classification

We propose a 3D-SaSiResNet to extract high-level features for supervised classification of the multiband images. Supposing the training set comprises N cases and can be represented by $\{\boldsymbol{s}_i, l_i\}, i = 1, 2, \ldots, N$, where $\boldsymbol{s}_i \in \mathbb{R}^{b_0}$ refers to the spectral signature of the ith training sample and l_i is the class label, the 3D spatial-adaptive patch $\boldsymbol{D}_i \in \mathbb{R}^{w \times w \times b_0}$ of $\boldsymbol{s}_i$ is extracted from the original multiband images data $\boldsymbol{I}_0 \in \mathbb{R}^{m_0 \times n_0 \times b_0}$ by reference to the superpixels generated in Section 2.1, where w is the window size of $\boldsymbol{D}_i$. The 3D patches of training samples are grouped in $C_N^2 = N(N-1)/2$ pairs $\{\boldsymbol{D}_i, \boldsymbol{D}_j\}, i \neq j, i, j = 1, 2, \ldots, N$, where C_N^2 refers to a combination of the N patches in mathematics, and it involves the number of times to select two patches from a set of N patches without repetition. The pairs $\{\boldsymbol{D}_i, \boldsymbol{D}_j\}$ are then fed into the two subnetworks of the proposed method. In other words, $N(N-1)/2$ pairs of 3D patches are compared when we train the network. The architecture of the 3D-SaSiResNet is illustrated in Figure 5, which contains two identical subnetworks with the same configurations and shared weights. The training procedure aims to train a model to discriminate between a series of pairs from same/different classes. As shown in Figure 5, the first subnetwork

takes $\boldsymbol{D}_i$ as input, followed by a sequence of 3D convolution [39], 3D batch normalization (3D-BN) [50], 3D rectified linear unit (3D-ReLU) [51], 3D max-pooling [52], and a 3D residual basic block [34] (or 3D ResNeXt basic block [53]) with skip connection, the output of the last pooling layer is flattened as a vector to connect with a fully connected layer, and the last vector $f(\boldsymbol{D}_i)$ is the features of $\boldsymbol{D}_i$ produced by the above-mentioned subnetwork. The second subnetwork takes $\boldsymbol{D}_j$ as input, followed by the same procedure as performed in the first subnetwork, and outputs the features $f(\boldsymbol{D}_j)$. In greater detail, the five kinds of layers involved in the 3D-SaSiResNet are described in the following:

- ***3D convolution layers***: The 3D convolution layers [39] directly take 3D cubes (e.g., 3D patch $\boldsymbol{D}_i$) as input and are more suitable for simultaneously extracting spectral-spatial features of the multiband images than the 2D ones. Each input 3D feature map of the convolution layer is convolved with a 3D filter and is then passed through an activation function to generate the output 3D feature map
$$\boldsymbol{D}^h = \boldsymbol{D}^{h-1} * \boldsymbol{K}^h + \boldsymbol{\beta}^h \tag{4}$$
where $\boldsymbol{D}^{h-1}$ and $\boldsymbol{D}^h$ are the feature maps of the $(h-1)$th and hth layers, $\boldsymbol{K}^h$ and $\boldsymbol{\beta}^h$ represent the 3D filter and bias in the hth layer, respectively.
- ***3D-BN layers***: The 3D-BN layers [50] reduce the internal covariate shift in the network model by normalization, which allows a more independent learning process in each layer. 3D-BN can regularize and accelerate the learning process, imposing a Gaussian distribution on the feature maps. Specifically, letting $\boldsymbol{V}$ be the original batch feature cube, 3D-BN produces the transformed output $\hat{\boldsymbol{V}}$ by
$$\hat{\boldsymbol{V}} = \frac{\boldsymbol{V} - MEAN(\boldsymbol{V})}{\sqrt{VAR(\boldsymbol{V}) + \epsilon}} * \gamma + \eta \tag{5}$$
where $MEAN(\cdot)$ and $VAR(\cdot)$ are the mean and standard-deviation calculated per-dimension over the mini-batches, respectively, γ and η are learnable parameter vectors, and $\epsilon > 0$ is a small number to prevent numerical instability.
- ***ReLU layers***: ReLU layers [51] are nonlinear activation functions applied to each element of the feature map to gain nonlinear representations. As a widely-used activation function in neural networks, ReLU is faster than other saturating activation functions. The formula of the ReLU function is
$$ReLU(x) = \max(0, x) \tag{6}$$
- ***3D pooling layers***: 3D pooling layers [52] reduce the data variance and computation complexity by sub-sampling operations, such as max-pooling and average-pooling. The feature maps are first divided into several non-overlapping 3D cubes, and then the maximum or average value within each 3D cube are mapped into the output feature map.
- ***3D residual basic blocks/3D ResNeXt basic blocks***: 3D residual basic blocks [34] (or 3D ResNeXt basic blocks [53]) are adopted in the proposed model to alleviate the vanishing/exploding gradient problem. Two types of connections are involved in the 3D residual basic blocks. The first is a feedforward connection that connects layer by layer, i.e., each layer is connected with the former layer and the next layer. The second is the shortcut connection that connects the input with its output and preserves information across layers. The outputs of those two types of connection are summed, followed by a ReLU layer
$$\begin{aligned} \hat{\boldsymbol{D}}^{h-1} &= \mathcal{I}(\boldsymbol{D}^{h-1}) + \mathcal{F}(\boldsymbol{D}^{h-1}, \mathcal{W}^{h-1}) \\ \boldsymbol{D}^h &= ReLU(\hat{\boldsymbol{D}}^{h-1}) \end{aligned} \tag{7}$$
where $\hat{\boldsymbol{D}}^{h-1}$ denotes the sum of the identity mapping $\mathcal{I}(\boldsymbol{D}^{h-1})$ and residual function $\mathcal{F}(\boldsymbol{D}^{h-1}, \mathcal{W}^{h-1})$, $\mathcal{W}^{h-1}$ is the weight matrix, and $\boldsymbol{D}^{h-1}$ and $\boldsymbol{D}^h$ are the input and output feature maps of the residual basic block. The structure of 3D ResNeXt basic blocks is similar to the 3D

residual basic blocks except for substituting the second convolution layer of the 3D residual basic block into grouped convolutions.

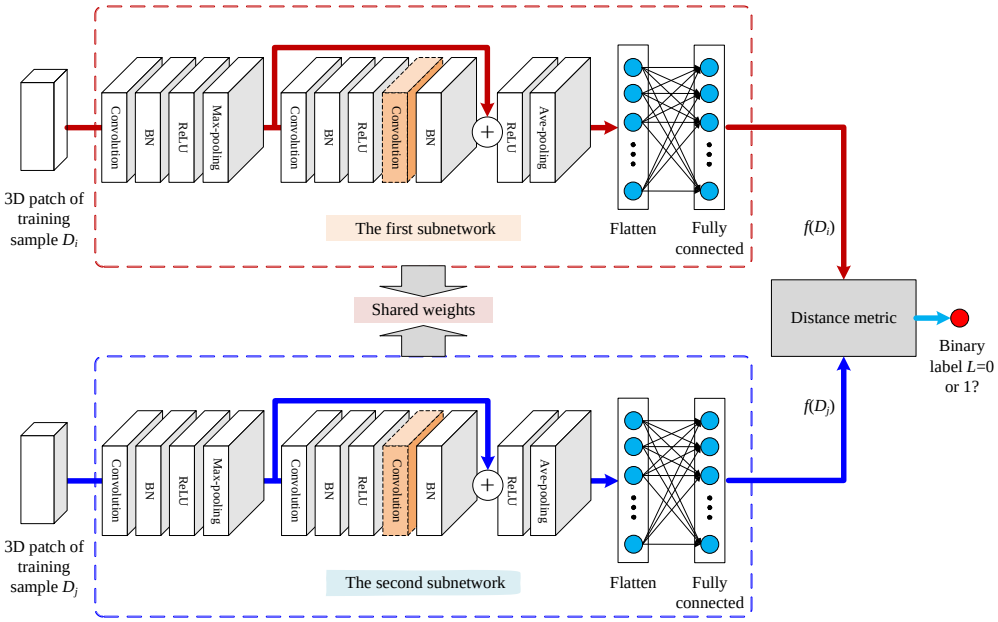


Figure 5. Architecture of the 3D-SaSiResNet. The 3D residual basic block in this architecture can also be replaced by a 3D ResNeXt basic block, in which the 3D convolution layer highlighted with an orange dotted cube should be changed to grouped convolutions.

Given the features $f(\boldsymbol{D}_i)$ and $f(\boldsymbol{D}_j)$ extracted by the first and second subnetworks, the following Euclidean distance is adopted as the metric to compute the distance between $\boldsymbol{D}_i$ and $\boldsymbol{D}_j$

$$E(\boldsymbol{D}_i, \boldsymbol{D}_j) = \|f(\boldsymbol{D}_i) - f(\boldsymbol{D}_j)\|_2 \tag{8}$$

After the architecture of the 3D-SaSiResNet is constructed, we train the model for many epochs using the pairs $\{\boldsymbol{D}_i, \boldsymbol{D}_j\}, i \neq j, i, j = 1, 2, \ldots, N$. The parameter $\boldsymbol{\theta}$ of the 3D-SaSiResNet is updated through minimizing the contrastive loss function $\mathcal{Q}(\boldsymbol{\theta})$

$$\begin{aligned} \mathcal{Q}(\boldsymbol{\theta}) &= \sum_{i=1,j=1,i\neq j}^{N} \mathcal{Q}(\boldsymbol{\theta}, (L, \boldsymbol{D}_i, \boldsymbol{D}_j)) \\ &= \sum_{i=1,j=1,i\neq j}^{N} (1-L)\mathcal{Q}_S(E(\boldsymbol{D}_i, \boldsymbol{D}_j)) + L\mathcal{Q}_D(E(\boldsymbol{D}_i, \boldsymbol{D}_j)) \\ &= \sum_{i=1,j=1,i\neq j}^{N} \frac{(1-L)}{2}(E(\boldsymbol{D}_i, \boldsymbol{D}_j))^2 + \frac{L}{2}\max(0, M - E(\boldsymbol{D}_i, \boldsymbol{D}_j))^2 \end{aligned} \tag{9}$$

where $(L, \boldsymbol{D}_i, \boldsymbol{D}_j)$ is the training sample pair, L is a binary label assigned to the pair $\{\boldsymbol{D}_i, \boldsymbol{D}_j\}$, $L = 0$ if $\boldsymbol{D}_i$ and $\boldsymbol{D}_j$ have the same label and $L = 1$ if $\boldsymbol{D}_i$ and $\boldsymbol{D}_j$ have different labels, $\mathcal{Q}_S$ and $\mathcal{Q}_D$ are designed to minimize $\mathcal{Q}$ with respect to $\boldsymbol{\theta}$ which will result in low values of $E(\boldsymbol{D}_i, \boldsymbol{D}_j)$ for similar pairs and high values of $E(\boldsymbol{D}_i, \boldsymbol{D}_j)$ for dissimilar pairs, and $M > 0$ is a margin. We set $M = 2$ in this study. Moreover, we use the adaptive moment estimation (Adam) [54] to minimize the loss function $\mathcal{Q}(\boldsymbol{\theta})$ and obtain the optimal parameters of the network.

Letting $\tilde{\boldsymbol{D}} \in \mathbb{R}^{w \times w \times b_0}$ be the 3D patch of a testing sample $\tilde{\boldsymbol{s}}$, the class label of $\tilde{\boldsymbol{s}}$ can be obtained by a nearest neighbor classifier. That is, $\tilde{\boldsymbol{D}}$ is independently and sequentially compared with each

$\boldsymbol{D}_i, i = 1, 2, \ldots, N$ and the class label of $\tilde{\boldsymbol{s}}$ is set to be the same as that of a specific training sample, whose features deserve the following minimal Euclidean distance:

$$\text{class } \tilde{\boldsymbol{s}} = \arg \min_{l_i, i=1,2,\ldots,N} \|f(\tilde{\boldsymbol{D}}) - f(\boldsymbol{D}_i)\|_2 \tag{10}$$

where $f(\tilde{\boldsymbol{D}})$ and $f(\boldsymbol{D}_i), i = 1, 2, \ldots, N$ are the features extracted by the trained network.

3. Experimental Results

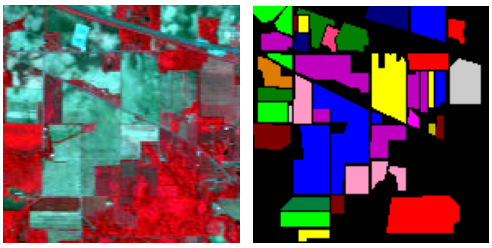
3.1. Dataset Description

To verify the performance of the proposed method, experiments are performed on three real-world multiband images datasets (i.e., Indian Pines data, University of Pavia data, and Zurich data). The following provides a brief description of the three datasets:

1. ***Indian Pines data***: the first dataset was gathered by the National Aeronautics and Space Administration's (NASA) Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) sensor in 1992 over an agricultural area of the Indian Pines test site in northwestern Indiana. The selected scene consists of 145 × 145 pixels and 220 spectral reflectance bands cover the wavelength range of 400 to 2500 nm. After removing 20 bands (i.e., bands 104–108, 150–163 and 220) corrupted by the atmospheric water-vapor absorption effect, 200 bands are considered for experiments. The spatial resolution of this dataset is about 20 m per pixel. The three-band false color composite image together with its corresponding ground truth is depicted in Figure 6. Sixteen different classes of land-covers are considered in the ground truth, which contains 10,366 labeled pixels ranging unbalanced from 20 to 2468, posing a big challenge to the classification problem. The number of training, validation, and testing samples in each class is shown in Table 1, whose background color corresponds to different classes of land-covers. Moreover, note that the total number of samples in class 9 (i.e., Oats) are only 20, and we set the number of validation samples to 5 in this class.
2. ***University of Pavia data***: the second dataset was collected by the Reflective Optics System Imaging Spectrometer (ROSIS) over the urban area surrounding the University of Pavia, south of Italy, on 8 July 2002. The dataset contains 103 hyperspectral bands covering the range from 430 to 860 nm and a scene made of 610 × 340 with a spatial resolution of 1.3 m per pixel is considered in the experiment. The three-band false color composite of this data and its corresponding ground truth are shown in Figure 7. We consider nine classes of land-covers in this dataset, and the number of training, validation, and testing samples in each class is shown in Table 2, whose background color distinguishes different classes of land-covers.
3. ***Zurich data***: The third dataset was captured by the QuickBird satellite in August 2002 over the city of Zurich, in Switzerland [55]. There are 20 images in the whole dataset and we choose the 18th image for experiments. The chosen image consists of four spectral bands spanning near-infrared to visible spectrum (NIR-R-G-B) and the size of each band is 748 × 800 with a spatial resolution of 0.61 m per pixel. The color composite image and the ground truth are plotted in Figure 8. Six classes of interest are contained in this dataset and the detailed number of training, validation and testing samples in each class is displayed in Table 3, whose background color represents different classes of land-covers.

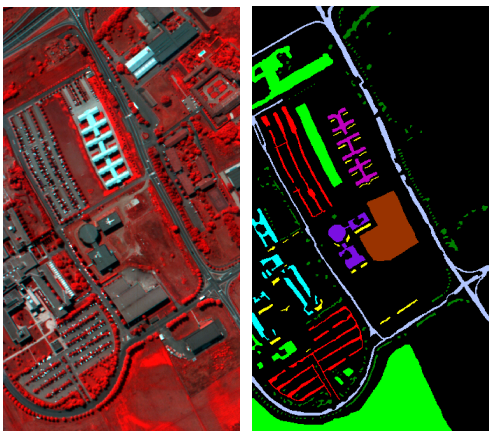
Table 1. Number of training, validation, and testing samples used in the Indian Pines data.

Class	Name	Training	Validation	Testing	Total
1	Alfalfa	10	10	34	54
2	Corn-no till	10	10	1414	1434
3	Corn-min till	10	10	814	834
4	Corn	10	10	214	234
5	Grass/pasture	10	10	477	497
6	Grass/trees	10	10	727	747
7	Grass/pasture-mowed	10	10	6	26
8	Hay-windrowed	10	10	469	489
9	Oats	10	5	5	20
10	Soybean-no till	10	10	948	968
11	Soybean-min till	10	10	2448	2468
12	Soybean-clean till	10	10	594	614
13	Wheat	10	10	192	212
14	Woods	10	10	1274	1294
15	Bldg-grass-tree-drives	10	10	360	380
16	Stone-steel towers	10	10	75	95
	Total	160	155	10,051	10,366



(**a**) (**b**)

Figure 6. Indian Pines data. (**a**) three-band false color composite, (**b**) ground truth data with 16 classes.

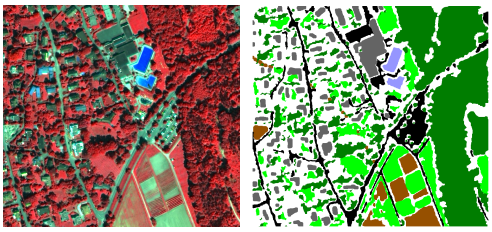


(**a**) (**b**)

Figure 7. University of Pavia data. (**a**) three-band false color composite, (**b**) ground truth data with nine classes.

Table 2. Number of training, validation, and testing samples used in the University of Pavia data.

Class	Name	Training	Validation	Testing	Total
1	Asphalt	10	10	6611	6631
2	Meadows	10	10	18,629	18,649
3	Gravel	10	10	2079	2099
4	Trees	10	10	3044	3064
5	Metal sheets	10	10	1325	1345
6	Bare soil	10	10	5009	5029
7	Bitumen	10	10	1310	1330
8	Bricks	10	10	3662	3682
9	shadows	10	10	927	947
	Total	90	90	42,596	42,776



(**a**) (**b**)

Figure 8. Zurich data. (**a**) three-band false color composite, (**b**) ground truth data with six classes.

Table 3. Number of training, validation, and testing samples used in the Zurich data.

Class	Name	Training	Validation	Testing	Total
1	Roads	10	10	51,162	51,182
2	Trees	10	10	185,116	185,136
3	Grass	10	10	65,616	65,636
4	Buildings	10	10	40,214	40,234
5	Bare soil	10	10	19,925	19,945
6	Swimming pools	10	10	4385	4405
	Total	60	60	366,418	366,538

3.2. Experimental Settings

The purpose of the experiments is to assess the effectiveness of the 3D spatial-adaptive patch construction method and the 3D-SaSiResNet-based classification method. To that end, we utilize the 2D-CNN, 2D-ResNet, 2D Siamese ResNet (abbreviated as 2D-SiResNet), 3D-CNN, 3D-ResNet, and the proposed 3D-SaSiResNet as classifiers, and compare the patches generated in a spatially adaptive fashion with the patches obtained by fixing the training (or validation, testing) samples at the center. Moreover, the SVM [13] by using spectral features is also compared with the 3D-SaSiResNet. As will be shown in Section 3.3, a total of 12 methods are given for comparative study by combining the patch construction and classification methods in pairs. In different methods, the abbreviations containing "Sa" denote **s**patial-**a**daptive-based methods, "-fix" denotes 2D or 3D patches are extracted by **fix**ing a certain sample at the center of neighborhood windows, "Si" denotes **Si**amese-based methods, "−1" denotes a 2D or 3D residual basic block that is used in the ResNet-based methods, "−2" denotes a 2D or 3D ResNeXt basic block that is used in the ResNet-based methods.

The architectures of the above-mentioned deep learning methods are displayed in Tables 4–9. It is notable that small kernels are effective choices for 2D-CNN [56] and 3D-CNN [39], and we fix the kernel sizes of all the 2D convolutions to 3×3, and most of the 3D convolutions to $3 \times 3 \times 3$. To speed up the 3D-based algorithms and consider more bands at a time, the kernel sizes of the first 3D convolution layers for Indian Pines data, University of Pavia data, and Zurich data are set to

$100 \times 3 \times 3$, $50 \times 3 \times 3$ and $3 \times 3 \times 3$, respectively. To make a fair comparison, all the deep learning methods are configured as similar to each other as possible. For instance, the subnetwork in the Siamese-based methods (i.e., 2D-SaSiResNet-1/2D-SaSiResNet-2/2D-SiResNet-2-fix and 3D-SaSiResNet-1/3D-SaSiResNet-2/3D-SiResNet-2-fix) are set up identical to the corresponding non-Siamese ones (i.e., 2D-SaCNN/2D-SaResNet-1/2D-SaResNet-2 and 3D-SaCNN/3D-SaResNet-1/3D-SaResNet-2), and the number of layers in the 2D-based networks are the same as the 3D-based ones. For the parameter settings, the intrinsic dimensionalities are calculated to be 5, 4, and 2 in the Indian Pines data, University of Pavia data, and Zurich data, respectively, and the number of superpixels are set to 589, 2602, and 19,543 in the three multiband images datasets, respectively. The patch size w of all the three experimental datasets is set to 9. In other words, each sample is represented by its 9×9 neighborhoods in 2D-SiResNet-2-fix and 3D-SiResNet-2-fix, while all the spatial-adaptive patches are resized to 9×9 so as to represent the objects of the multiband images. The reasons for resizing the patches to 9×9 are that, if w is too small (e.g., 3×3), the spatial information will not be adequately learned by the network; on the contrary, if w is too large (e.g., 15×15), the computational burden will be increased and the training process will slow down. The batch size and training epochs are taken as 20 and 100, respectively. Furthermore, the Adam solver is adopted to train the networks and the learning rate is initialized to 0.0001.

Table 4. Detailed configuration of the 2D-SaCNN. Bold type indicates the number of feature maps in each layer.

Layer	Input Size	Kernel Size	Stride	Padding	Cardinality	Output Size
Convolution-1	$\mathbf{1} \times w \times w \times b_0$	3×3	(1,1)	(0,0)	1	$\mathbf{32} \times (w-2) \times (w-2)$
BN-1	$\mathbf{32} \times (w-2) \times (w-2)$	–	–	–	–	$\mathbf{32} \times (w-2) \times (w-2)$
ReLU-1	$\mathbf{32} \times (w-2) \times (w-2)$	–	–	–	–	$\mathbf{32} \times (w-2) \times (w-2)$
Max-pooling-1	$\mathbf{32} \times (w-2) \times (w-2)$	3×3	(2,2)	(0,0)	–	$\mathbf{32} \times w_1$ [1] $\times w_1$
Convolution-2	$\mathbf{32} \times w_1 \times w_1$	3×3	(1,1)	(1,1)	1	$\mathbf{64} \times w_1 \times w_1$
BN-2	$\mathbf{64} \times w_1 \times w_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1$
ReLU-2	$\mathbf{64} \times w_1 \times w_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1$
Convolution-3	$\mathbf{64} \times w_1 \times w_1$	3×3	(1,1)	(1,1)	1	$\mathbf{64} \times w_1 \times w_1$
BN-3	$\mathbf{64} \times w_1 \times w_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1$
ReLU-3	$\mathbf{64} \times w_1 \times w_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1$
Flatten	$\mathbf{64} \times w_1 \times w_1$	–	–	–	–	$[\mathbf{64} \times w_1 \times w_1] \times 1$
Output	$[\mathbf{64} \times w_1 \times w_1] \times 1$	–	–	–	–	C [2] $\times 1$

[1] $w_1 = \frac{(w-2)-3}{2} + 1$. [2] C denotes the number of land-covers in the multiband image.

Moreover, the SVM and 3D spatial-adaptive patches generation procedure is performed by MATLAB R2019a (The MathWorks, United States) on a Windows 10 operating system (Microsoft, United States) and the deep learning-based classification methods are executed on a server with Nvidia Tesla K80 GPU platform (NVIDIA, United States) and Pytorch backend. In all three datasets, very limited labeled samples, i.e., 10 samples per class, are randomly chosen as training samples, 10 (or 5 samples in class 9 (i.e., Oats) of the Indian Pines data) per class are for validation and the rest are for testing. To study the influence of the number of training samples, we also choose 20 and 50 samples from each class for training. Since classes 1 (i.e., Alfalfa), 7 (i.e., Grass/pasture-mowed), and 9 (i.e., Oats) contain very few samples for the Indian Pines data, the number of training samples in those classes is constantly set to 10 in different cases. The number of validation samples is the same as that in Tables 1–3. The label of a training patch is set to be the same as that of most training samples in this patch. In case the training patches of a certain class are less than 2, we perform rotation augmentation, whose rotation angle is 90 degrees, to augment the number of training patches to 2. Three popular indices, i.e., overall accuracy (OA), average accuracy (AA), and kappa coefficient (κ), are computed to compare different methods quantitatively. All three of the indices are obtained by comparing the reference labels and the calculated labels of test samples. Each experiment is repeated

ten times using a repeated random subsampling validation strategy to alleviate possible bias and the average results are displayed in the tables.

Table 5. Detailed configuration of the 2D-SaResNet-1 and a subnetwork of the 2D-SaSiResNet-1. Bold type indicates the number of feature maps in each layer.

Layer	Input Size	Kernel Size	Stride	Padding	Cardinality	Output Size
Convolution-1	$\mathbf{1} \times w \times w \times b_0$	3×3	(1,1)	(0,0)	1	$\mathbf{32} \times (w-2) \times (w-2)$
BN-1	$\mathbf{32} \times (w-2) \times (w-2)$	–	–	–	–	$\mathbf{32} \times (w-2) \times (w-2)$
ReLU-1	$\mathbf{32} \times (w-2) \times (w-2)$	–	–	–	–	$\mathbf{32} \times (w-2) \times (w-2)$
Max-pooling-1	$\mathbf{32} \times (w-2) \times (w-2)$	3×3	(2,2)	(0,0)	–	$\mathbf{32} \times w_1$ [1] $\times w_1$
Convolution-2	$\mathbf{32} \times w_1 \times w_1$	3×3	(1,1)	(1,1)	1	$\mathbf{64} \times w_1 \times w_1$
BN-2	$\mathbf{64} \times w_1 \times w_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1$
ReLU-2	$\mathbf{64} \times w_1 \times w_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1$
Convolution-3	$\mathbf{64} \times w_1 \times w_1$	3×3	(1,1)	(1,1)	1	$\mathbf{64} \times w_1 \times w_1$
BN-3	$\mathbf{64} \times w_1 \times w_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1$
Shortcut	*Output of Max-pooling 1 + Output of BN 3*					
ReLU-3	$\mathbf{64} \times w_1 \times w_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1$
Flatten	$\mathbf{64} \times w_1 \times w_1$	–	–	–	–	$[\mathbf{64} \times w_1 \times w_1] \times 1$
Output	$[\mathbf{64} \times w_1 \times w_1] \times 1$	–	–	–	–	C [2] $\times 1$

[1] $w_1 = \frac{(w-2)-3}{2} + 1$. [2] C denotes the number of land-covers in the multiband image.

Table 6. Detailed configuration of the 2D-SaResNet-2, and a subnetwork of the 2D-SaSiResNet-2 and 2D-SiResNet-2-fix. Bold type indicates the number of feature maps in each layer.

Layer	Input Size	Kernel Size	Stride	Padding	Cardinality	Output Size
Convolution-1	$\mathbf{1} \times w \times w \times b_0$	3×3	(1,1)	(0,0)	8	$\mathbf{32} \times (w-2) \times (w-2)$
BN-1	$\mathbf{32} \times (w-2) \times (w-2)$	–	–	–	–	$\mathbf{32} \times (w-2) \times (w-2)$
ReLU-1	$\mathbf{32} \times (w-2) \times (w-2)$	–	–	–	–	$\mathbf{32} \times (w-2) \times (w-2)$
Max-pooling-1	$\mathbf{32} \times (w-2) \times (w-2)$	3×3	(2,2)	(0,0)	–	$\mathbf{32} \times w_1$ [1] $\times w_1$
Convolution-2	$\mathbf{32} \times w_1 \times w_1$	3×3	(1,1)	(1,1)	8	$\mathbf{64} \times w_1 \times w_1$
BN-2	$\mathbf{64} \times w_1 \times w_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1$
ReLU-2	$\mathbf{64} \times w_1 \times w_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1$
Convolution-3	$\mathbf{64} \times w_1 \times w_1$	3×3	(1,1)	(1,1)	8	$\mathbf{64} \times w_1 \times w_1$
BN-3	$\mathbf{64} \times w_1 \times w_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1$
Shortcut	*Output of Max-pooling 1 + Output of BN 3*					
ReLU-3	$\mathbf{64} \times w_1 \times w_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1$
Flatten	$\mathbf{64} \times w_1 \times w_1$	–	–	–	–	$[\mathbf{64} \times w_1 \times w_1] \times 1$
Output	$[\mathbf{64} \times w_1 \times w_1] \times 1$	–	–	–	–	C [2] $\times 1$

[1] $w_1 = \frac{(w-2)-3}{2} + 1$. [2] C denotes the number of land-covers in the multiband image.

Table 7. Detailed configuration of the 3D-SaCNN. Bold type indicates the number of feature maps in each layer.

Layer	Input Size	Kernel Size	Stride	Padding	Cardinality	Output Size
Convolution-1	$\mathbf{1} \times w \times w \times b_0$	$d \times 3 \times 3$	(1,1,1)	(0,0,0)	1	$\mathbf{32} \times (w-2) \times (w-2) \times (b_0-d+1)$
BN-1	$\mathbf{32} \times (w-2) \times (w-2) \times (b_0-d+1)$	–	–	–	–	$\mathbf{32} \times (w-2) \times (w-2) \times (b_0-d+1)$
ReLU-1	$\mathbf{32} \times (w-2) \times (w-2) \times (b_0-d+1)$	–	–	–	–	$\mathbf{32} \times (w-2) \times (w-2) \times (b_0-d+1)$
Max-pooling-1	$\mathbf{32} \times (w-2) \times (w-2) \times (b_0-d+1)$	$3 \times 3 \times 3$	(2,2,2)	(0,0,0)	–	$\mathbf{32} \times w_1 \times w_1 \times b_1$
Convolution-2	$\mathbf{32} \times w_1 \times w_1 \times b_1$	$3 \times 3 \times 3$	(1,1,1)	(1,1,1)	1	$\mathbf{64} \times w_1 \times w_1 \times b_1$
BN-2	$\mathbf{64} \times w_1 \times w_1 \times b_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1 \times b_1$
ReLU-2	$\mathbf{64} \times w_1 \times w_1 \times b_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1 \times b_1$
Convolution-3	$\mathbf{64} \times w_1 \times w_1 \times b_1$	$3 \times 3 \times 3$	(1,1,1)	(1,1,1)	1	$\mathbf{64} \times w_1 \times w_1 \times b_1$
BN-3	$\mathbf{64} \times w_1 \times w_1 \times b_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1 \times b_1$
ReLU-3	$\mathbf{64} \times w_1 \times w_1 \times b_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1 \times b_1$
Flatten	$\mathbf{64} \times w_1 \times w_1 \times b_1$	–	–	–	–	$[\mathbf{64} \times w_1 \times w_1 \times b_1] \times 1$
Output	$[\mathbf{64} \times w_1 \times w_1 \times b_1] \times 1$	–	–	–	–	$C \times 1$

[1] $w_1 = \frac{(w-2)-3}{2} + 1$. [2] $b_1 = \frac{(b_0-d+1)-3}{2} + 1$. [3] C denotes the number of land-covers in the multiband image.

Table 8. Detailed configuration of the 3D-SaResNet-1 and a subnetwork of the 3D-SaSiResNet-1. Bold type indicates the number of feature maps in each layer.

Layer	Input Size	Kernel Size	Stride	Padding	Cardinality	Output Size
Convolution-1	$\mathbf{1} \times w \times w \times b_0$	$d \times 3 \times 3$	(1,1,1)	(0,0,0)	1	$\mathbf{32} \times (w-2) \times (w-2) \times (b_0-d+1)$
BN-1	$\mathbf{32} \times (w-2) \times (w-2) \times (b_0-d+1)$	–	–	–	–	$\mathbf{32} \times (w-2) \times (w-2) \times (b_0-d+1)$
ReLU-1	$\mathbf{32} \times (w-2) \times (w-2) \times (b_0-d+1)$	–	–	–	–	$\mathbf{32} \times (w-2) \times (w-2) \times (b_0-d+1)$
Max-pooling-1	$\mathbf{32} \times (w-2) \times (w-2) \times (b_0-d+1)$	$3 \times 3 \times 3$	(2,2,2)	(0,0,0)	–	$\mathbf{32} \times w_1 \times w_1 \times b_1$
Convolution-2	$\mathbf{32} \times w_1 \times w_1 \times b_1$	$3 \times 3 \times 3$	(1,1,1)	(1,1,1)	1	$\mathbf{64} \times w_1 \times w_1 \times b_1$
BN-2	$\mathbf{64} \times w_1 \times w_1 \times b_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1 \times b_1$
ReLU-2	$\mathbf{64} \times w_1 \times w_1 \times b_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1 \times b_1$
Convolution-3	$\mathbf{64} \times w_1 \times w_1 \times b_1$	$3 \times 3 \times 3$	(1,1,1)	(1,1,1)	1	$\mathbf{64} \times w_1 \times w_1 \times b_1$
BN-3	$\mathbf{64} \times w_1 \times w_1 \times b_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1 \times b_1$
Shortcut	*Output of Max-pooling 1 + Output of BN 3*					
ReLU-3	$\mathbf{64} \times w_1 \times w_1 \times b_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1 \times b_1$
Flatten	$\mathbf{64} \times w_1 \times w_1 \times b_1$	–	–	–	–	$[\mathbf{64} \times w_1 \times w_1 \times b_1] \times 1$
Output	$[\mathbf{64} \times w_1 \times w_1 \times b_1] \times 1$	–	–	–	–	$C \times 1$

[1] $w_1 = \frac{(w-2)-3}{2} + 1$. [2] $b_1 = \frac{(b_0-d+1)-3}{2} + 1$. [3] C denotes the number of land-covers in the multiband image.

Table 9. Detailed configuration of the 3D-SaResNet-2, and a subnetwork of the 3D-SaSiResNet-2 and 3D-SiResNet-2-fix. Bold type indicates the number of feature maps in each layer.

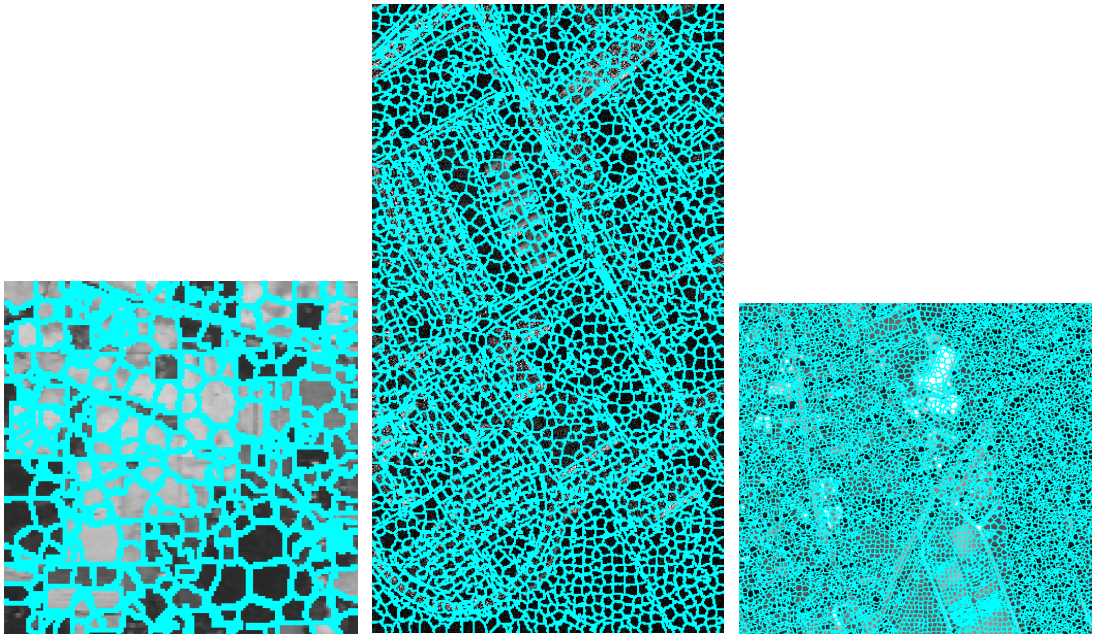
Layer	Input Size	Kernel Size	Stride	Padding	Cardinality	Output Size
Convolution-1	$\mathbf{1} \times w \times w \times b_0$	$d \times 3 \times 3$	(1,1,1)	(0,0,0)	8	$\mathbf{32} \times (w-2) \times (w-2) \times (b_0-d+1)$
BN-1	$\mathbf{32} \times (w-2) \times (w-2) \times (b_0-d+1)$	–	–	–	–	$\mathbf{32} \times (w-2) \times (w-2) \times (b_0-d+1)$
ReLU-1	$\mathbf{32} \times (w-2) \times (w-2) \times (b_0-d+1)$	–	–	–	–	$\mathbf{32} \times (w-2) \times (w-2) \times (b_0-d+1)$
Max-pooling-1	$\mathbf{32} \times (w-2) \times (w-2) \times (b_0-d+1)$	$3 \times 3 \times 3$	(2,2,2)	(0,0,0)	–	$\mathbf{32} \times w_1 \times w_1 \times b_1$
Convolution-2	$\mathbf{32} \times w_1 \times w_1 \times b_1$	$3 \times 3 \times 3$	(1,1,1)	(1,1,1)	8	$\mathbf{64} \times w_1 \times w_1 \times b_1$
BN-2	$\mathbf{64} \times w_1 \times w_1 \times b_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1 \times b_1$
ReLU-2	$\mathbf{64} \times w_1 \times w_1 \times b_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1 \times b_1$
Convolution-3	$\mathbf{64} \times w_1 \times w_1 \times b_1$	$3 \times 3 \times 3$	(1,1,1)	(1,1,1)	8	$\mathbf{64} \times w_1 \times w_1 \times b_1$
BN-3	$\mathbf{64} \times w_1 \times w_1 \times b_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1 \times b_1$
Shortcut	*Output of Max-pooling 1 + Output of BN 3*					
ReLU-3	$\mathbf{64} \times w_1 \times w_1 \times b_1$	–	–	–	–	$\mathbf{64} \times w_1 \times w_1 \times b_1$
Flatten	$\mathbf{64} \times w_1 \times w_1 \times b_1$	–	–	–	–	$[\mathbf{64} \times w_1 \times w_1 \times b_1] \times 1$
Output	$[\mathbf{64} \times w_1 \times w_1 \times b_1] \times 1$	–	–	–	–	$C \times 1$

[1] $w_1 = \frac{(w-2)-3}{2} + 1$. [2] $b_1 = \frac{(b_0-d+1)-3}{2} + 1$. [3] C denotes the number of land-covers in the multiband image.

3.3. Classification Results

The final superpixel segmentation results of the three experimental datasets are shown in Figure 9, from which we find that the proposed superpixel map generation method flexibly divides the multiband images data into naturally formed spatial areas with irregular sizes and shapes. Tables 10–12 report the class-specific accuracy, OA, AA, κ, and computation time of different methods, while the thematic maps are visually depicted in Figures 10–12. According to the reported results, a few observations are noteworthy. It can be first seen that, in most cases, the spatial-adaptive patches outperforms the patches generated by fixing the training/validation/testing samples at the center. As an example, for the Indian Pines data, the OA and κ of the 2D-SaSiResNet-2 are respectively 8.24% and 9.00% higher than the 2D-SiResNet-2-fix (see Table 10), while the improvement in OA of the 3D-SaSiResNet-2 is at least 3% more than its corresponding fixed center version (i.e., 3D-SiResNet-2-fix). We can observe from Figure 10 that the classification maps of 2D-SaSiResNet-2/3D-SaSiResNet-2 are much less noisy than those of the 2D-SiResNet-2-fix/3D-SiResNet-2-fix. For the University of Pavia data (see Table 11) and Zurich data (see Table 12), although the improvement in classification accuracy is not as much as the Indian Pines data, the methods with spatial-adaptive patches still provide better results than those with a fixed center in most situations. It is shown in Figure 11 that 2D-SiResNet-2-fix/3D-SiResNet-2-fix fail to correctly classify the pixels in some regions (e.g., the *Bricks* (class 8) located at the middle-left region), while 2D-SaSiResNet-2/3D-SaSiResNet-2 give a better classification performance than 2D-SiResNet-2-fix/3D-SiResNet-2-fix. Moreover, as shown in Figure 12, the classification maps of 2D-SiResNet-2-fix/3D-SiResNet-2-fix assign more areas to *Road* (class 1). This makes the classification accuracy of the *Road* (class 1) higher than 2D-SaSiResNet-2/3D-SaSiResNet-2; nevertheless, it sacrifices the accuracy of other land-covers.

The reason for good results of the spatial-adaptive patch construction method is that they can take into consideration the spatial structures and region boundary areas of the remote sensing data while its counterpart forcibly divides the data into fixed center patches, ignoring the heterogeneous land covers, especially those at the border of different classes.



(**a**) Indian Pines data (**b**) University of Pavia data (**c**) Zurich data

Figure 9. Final superpixel segmentation results of the three experimental datasets. (**a**) Indian Pines data, (**b**) University of Pavia data, and (**c**) Zurich data.

We then compare the 3D-SaSiResNet-1/3D-SaSiResNet-2 against other methods. It can be observed from Tables 10–12 that SVM and 2D-SaCNN perform the worst, the 3D-SaCNN, 2D-SaResNet-1/2D-SaResNet-2, and 3D-SaResNet-1/3D-SaResNet-2 have better performances than the SVM and 2D-SaCNN but comparable or slightly inferior to the 2D-SaSiResNet-1/2D-SaSiResNet-2, and 3D-SaSiResNet-1/3D-SaSiResNet-2 yield superior performance to the competing methods. As given in Table 10, the OA of the SVM and 2D-SaCNN are at least 10% lower than the 3D-SaCNN, 2D-SaResNet-1/2D-SaResNet-2 and 3D-SaResNet-1/3D-SaResNet-2 for the Indian Pines data. It can be obviously seen from Tables 11 and 12 that the classification performance of the SVM and 2D-SaCNN is also worse than the 3D-SaCNN, 2D-SaResNet-1/2D-SaResNet-2, and 3D-SaResNet-1/3D-SaResNet-2 for the other two datasets. One reason why 3D-SaCNN performs better than SVM and 2D-SaCNN is that the 3D convolution can simultaneously extract the spectral-spatial features, and the reason why 2D-SaResNet-1/2D-SaResNet-2 outperform 2D-SaCNN is mainly because shortcut connections are adopted to improve the effectiveness of the training procedure. Moreover, the Siamese-based methods perform much better than their corresponding no-Siamese versions. Specifically, it is notable from Table 10 that the 3D-SiSaResNet-2 provides the best performance in all the algorithms under comparison. Classification results of the University of Pavia data and Zurich data also reveal similar properties. This implies that the Siamese architecture of the network combined with 3D ResNet can effectively improve the classification performance. The main reason why the Siamese architecture is suitable for classification of multiband images with very few training samples is because the network is trained by inputting the training samples in pairs and thus the separability of different classes are learned and the features extracted are especially suitable for the classification task.

An additional noticeable point is that the 3D-based classification methods achieve comparable or better performance than the 2D-based ones. As illustrated in Table 11, in comparison with

the 2D-SaCNN, the OA of 3D-SaCNN is improved by 8.79% for the University of Pavia, while the OA of 3D-SaSiResNet-2 is 2.84% higher than its 2D scenario (i.e., 2D-SaSiResNet-2). Similarly, it can be found from Tables 10–12 that the 3D-SaCNN, 3D-SaResNet-1/3D-SaResNet-2, 3D-SaSiResNet-1/3D-SaSiResNet-2, and 3D-SiResNet-2-fix have better classification results than their corresponding 2D versions. Moreover, it is also clearly visible from Figures 10–12 that the classification maps of the 3D-based methods are much closer to the ground truth (see Figures 6b, 7b and 8b) than the 2D-based ones. This is due to the fact that, by representing the multiband images in 3D cubes, the joint spectral-spatial structures are effectively learned by the 3D-based classification methods.

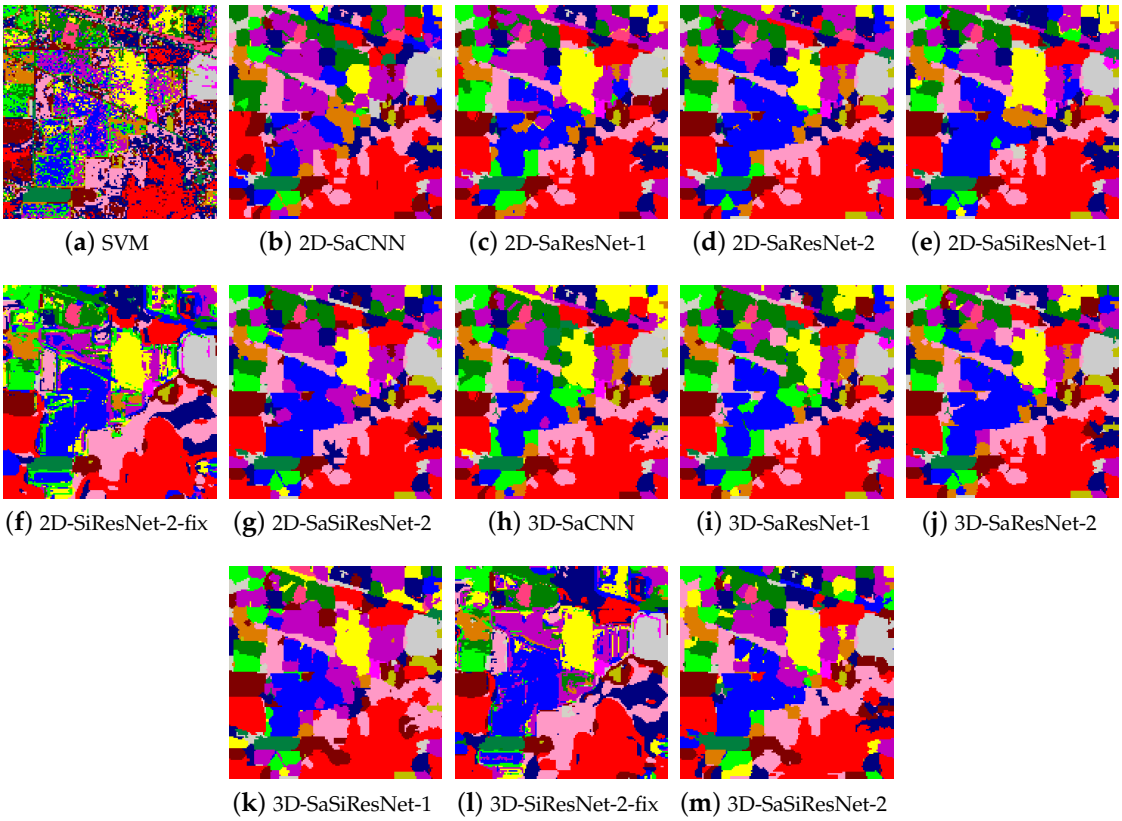
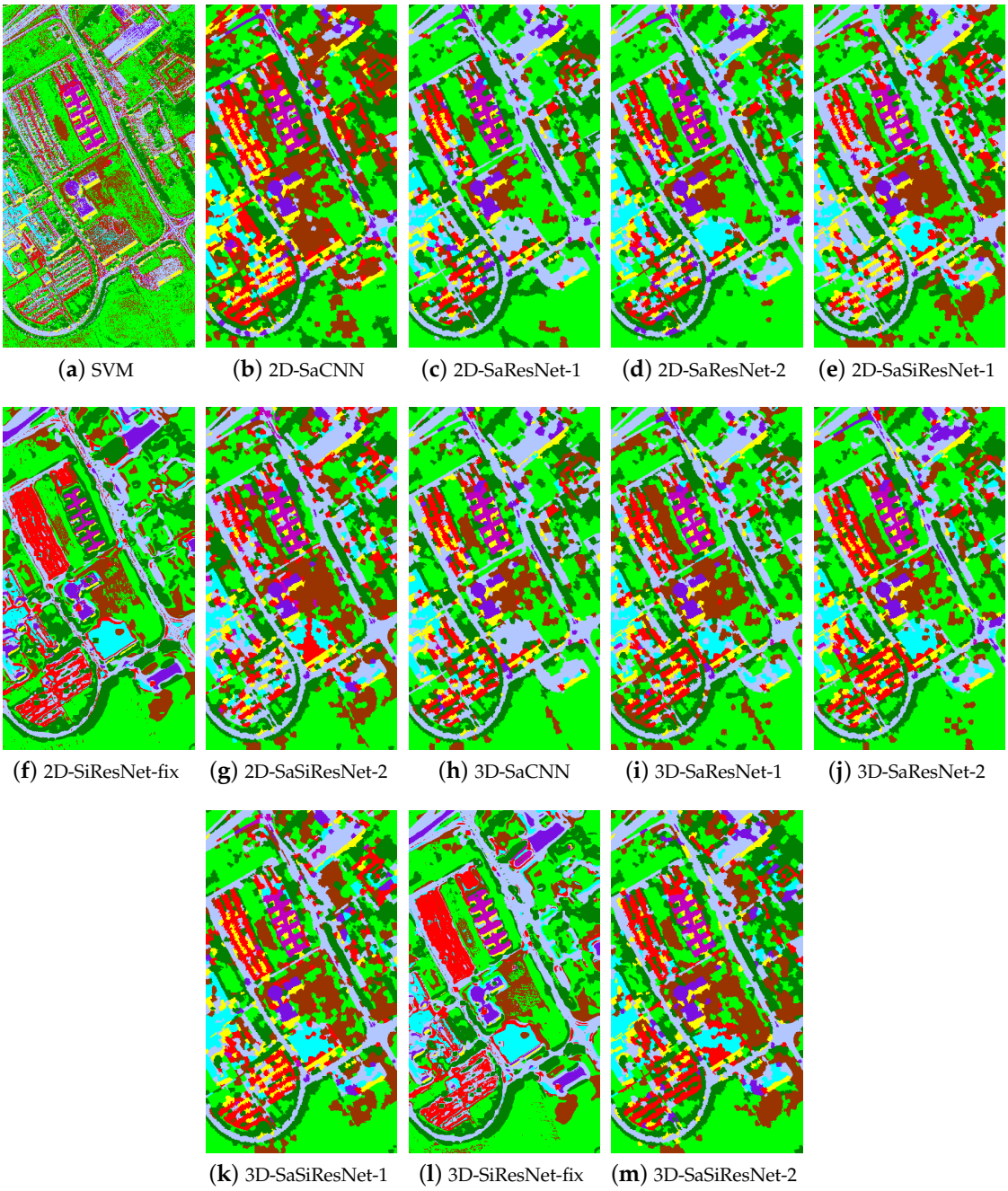


Figure 10. Classification maps of the Indian Pines data obtained by (**a**) SVM, (**b**) 2D-SaCNN, (**c**) 2D-SaResNet-1, (**d**) 2D-SaResNet-2, (**e**) 2D-SaSiResNet-1, (**f**) 2D-SiResNet-fix, (**g**) 2D-SaSiResNet-2, (**h**) 3D-SaCNN, (**i**) 3D-SaResNet-1, (**j**) 3D-SaResNet-2, (**k**) 3D-SaSiResNet-1, (**l**) 3D-SiResNet-2-fix and (**m**) 3D-SaSiResNet-2.

To further validate the effectiveness of the proposed method, we focus on the features learned by the last layer of various deep learning-based methods. It is notable from Table 10 that the land covers *Corn-no till* (class 2), *Corn-min till* (class 3), and *Soybean-no till* (class 10) are difficult-to-separate classes for the Indian Pines data. For instance, as for class 3, the classification accuracy of the 2D-SaCNN is as low as 43.21%, the 2D-SaSiResNet-2 and 3D-SaSiResNet-2 significantly improve the classification performance, and the classification accuracy of the 3D-SaSiResNet-2 is increased to 81.92%. As for class 2 and class 10, the proposed 3D-SaSiResNet-2 also provides better results than the state-of-the-art methods. To give an intuitive explanation, the two-dimensional features of the Indian Pines data obtained by 2D-SaCNN, 2D-SaSiResNet-2, 3D-SiResNet-2-fix, and 3D-SaSiResNet-2 in a trial are plotted in Figure 13, from which we can see that the representations learned by the 3D-SaSiResNet-2 are more scattered and separable than other methods. Therefore, it is much easier to distinguish the samples with the features obtained by the 3D-SaSiResNet-2. Moreover, as can be observed from Tables 10–12, although 3D-SaSiResNet-1 and 3D-SaSiResNet-2 take more computation

time than no-Siamese-based methods, they are much more efficient than the 2D-SiResNet-2-fix and 3D-SiResNet-2-fix for the Indian Pines data and Zurich data. As to the University of Pavia data, although the processing time of 3D-SaSiResNet-1/3D-SaSiResNet-2 is about four minutes longer than 2D-SiResNet-2-fix/3D-SiResNet-2-fix, they can complete the classification task in less than 20 min. 3D-SaSiResNet-1/3D-SaSiResNet-2 are efficient because they are object-based methods rather than pixel-based methods, and the kernel sizes of the first 3D convolution layers are set to $100 \times 3 \times 3$ and $50 \times 3 \times 3$ in the two hyperspectral datasets rather than $3 \times 3 \times 3$. In short, the experimental results validate the effectiveness of 3D-SaSiResNet in the classification of multiband images data.



(**a**) SVM (**b**) 2D-SaCNN (**c**) 2D-SaResNet-1 (**d**) 2D-SaResNet-2 (**e**) 2D-SaSiResNet-1

(**f**) 2D-SiResNet-fix (**g**) 2D-SaSiResNet-2 (**h**) 3D-SaCNN (**i**) 3D-SaResNet-1 (**j**) 3D-SaResNet-2

(**k**) 3D-SaSiResNet-1 (**l**) 3D-SiResNet-fix (**m**) 3D-SaSiResNet-2

Figure 11. Classification maps of the University of Pavia data obtained by (**a**) SVM, (**b**) 2D-SaCNN, (**c**) 2D-SaResNet-1, (**d**) 2D-SaResNet-2, (**e**) 2D-SaSiResNet-1, (**f**) 2D-SiResNet-fix, (**g**) 2D-SaSiResNet-2, (**h**) 3D-SaCNN, (**i**) 3D-SaResNet-1, (**j**) 3D-SaResNet-2, (**k**) 3D-SaSiResNet-1, (**l**) 3D-SiResNet-2-fix and (**m**) 3D-SaSiResNet-2.

Table 10. Classification accuracy (%) of various methods [1] for the Indian Pines data with 10 labeled training samples per class, bold values indicate the best result for a row.

Class	SVM	2D-Based Methods						3D-Based Methods					
		SaCNN	SaResNet-1	SaResNet-2	SaSiResNet-1	SiResNet-2-fix	SaSiResNet-2	SaCNN	SaResNet-1	SaResNet-2	SaSiResNet-1	SiResNet-2-fix	SaSiResNet-2
1	80.29	8.24	84.97	92.06	94.41	95.00	84.41	94.71	94.71	94.71	94.71	95.00	**95.88**
2	40.57	50.48	51.54	48.68	59.43	49.29	54.51	44.06	43.47	37.99	**66.85**	55.37	66.61
3	53.51	43.21	63.36	64.66	77.17	54.84	76.41	77.81	79.18	68.05	79.30	65.43	**81.92**
4	68.74	77.38	78.71	77.99	84.21	87.24	76.65	88.27	83.55	75.19	81.73	**88.41**	80.51
5	72.73	39.71	49.36	64.68	84.93	78.43	86.76	83.14	72.62	83.33	**90.17**	82.35	89.18
6	71.02	77.59	90.91	**95.25**	89.08	87.79	86.85	89.63	90.50	90.83	88.78	90.54	89.71
7	85.00	90.00	**100.00**	**100.00**	**100.00**	96.67	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	98.33	**100.00**
8	83.45	98.06	**99.57**	**99.57**	94.95	91.83	94.90	95.61	96.67	97.42	96.25	92.75	95.65
9	94.00	18.00	35.56	36.00	30.00	**100.00**	44.50	36.00	36.00	36.00	38.00	**100.00**	36.00
10	54.56	32.41	62.68	54.02	75.81	67.45	72.15	62.80	63.48	70.04	72.61	68.63	**77.68**
11	35.16	38.42	52.99	51.90	53.52	52.39	**65.60**	43.00	45.92	56.26	59.44	61.87	59.23
12	48.38	66.68	55.33	68.97	72.29	56.14	**79.29**	72.05	76.18	69.12	76.11	70.77	75.96
13	91.93	85.73	98.61	98.59	98.59	**98.96**	98.54	98.59	98.59	98.59	95.42	98.02	98.59
14	79.80	83.30	94.37	**95.64**	91.00	89.18	92.90	92.84	92.01	89.22	93.15	89.59	90.08
15	52.28	73.50	77.75	77.78	83.47	77.47	83.80	**86.75**	79.69	77.64	79.97	73.58	81.14
16	79.33	80.53	78.52	78.67	78.27	**99.60**	80.29	79.07	79.87	79.07	77.20	94.67	77.47
OA(%)	55.26	56.20	67.29	67.93	73.15	67.17	75.41	67.57	67.88	68.88	76.05	72.48	**76.33**
AA(%)	68.17	60.20	73.39	75.28	79.19	80.14	79.85	77.78	77.03	76.47	80.60	**82.83**	80.98
κ(%)	50.09	50.88	63.00	63.77	69.84	63.16	72.16	63.77	64.04	64.87	73.00	68.93	**73.33**
time(min) [2]	0 + 0.02	1.86 + **1.11**	1.86 + 1.59	1.86 + 1.54	1.86 + 14.25	**0.11** + 58.33	1.86 + 14.11	1.86 + 1.55	1.86 + 1.64	1.86 + 1.58	1.86 + 14.42	**0.11** + 76.47	1.86 + 14.31

[1] "Sa" denotes **s**patial-**a**daptive-based methods; "Si" denotes **Si**amese-based methods; "-fix" denotes 2D or 3D patches are extracted by **fix**ing a certain sample at the center of neighborhood windows.; "-1" denotes a 2D or 3D residual basic block is used in the ResNet-based methods; "-2" denotes a 2D or 3D ResNeXt basic block is used in the ResNet-based methods. [2] The time before "+" denotes the computation time of patch construction step, while the time after "+" denotes the computation time of classification step.

Table 11. Classification accuracy (%) of various methods [1] for the University of Pavia data with 10 labeled training samples per class, bold values indicate the best result for a row.

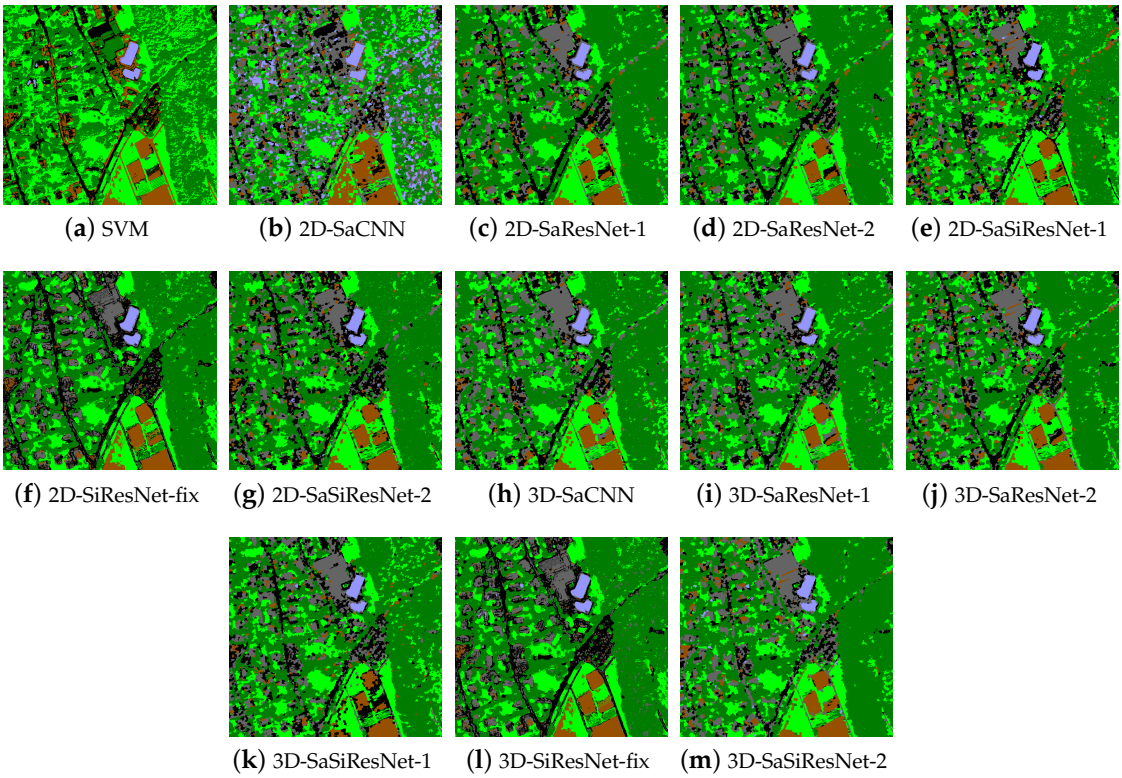
Class	SVM	2D-Based Methods						3D-Based Methods					
		SaCNN	SaResNet-1	SaResNet-2	SaSiResNet-1	SiResNet-2-fix	SaSiResNet-2	SaCNN	SaResNet-1	SaResNet-2	SaSiResNet-1	SiResNet-2-fix	SaSiResNet-2
1	66.83	64.19	80.45	84.73	82.90	74.12	83.78	81.15	86.79	85.64	87.23	88.12	**89.83**
2	67.36	62.34	80.27	82.78	76.82	**84.83**	76.34	73.35	74.93	76.34	75.83	78.04	79.12
3	65.63	**87.98**	82.33	80.01	82.38	73.87	76.18	79.69	72.59	78.07	80.43	84.13	84.29
4	87.26	92.40	**93.61**	93.18	91.50	91.60	91.14	90.93	88.85	88.57	89.63	90.65	90.78
5	96.39	97.92	96.77	95.25	94.27	97.82	93.51	97.67	97.35	**98.01**	95.55	96.52	95.56
6	50.92	52.61	44.29	46.37	73.86	67.10	**79.23**	70.46	72.11	66.80	78.12	75.46	73.80
7	72.24	95.31	96.40	95.53	93.75	85.20	93.90	93.83	96.18	96.00	**97.90**	97.75	97.60
8	53.44	66.42	64.90	62.29	62.49	65.24	71.12	64.80	67.99	71.92	**83.14**	79.87	79.57
9	89.05	83.60	81.27	78.99	77.72	**96.30**	74.95	80.58	80.81	80.50	79.34	78.92	77.13
OA(%)	67.01	67.82	76.83	78.35	78.59	80.02	79.48	76.61	78.20	78.61	81.08	81.86	**82.32**
AA(%)	72.13	78.09	80.03	79.90	81.74	81.79	82.24	81.38	81.96	82.43	85.24	**85.49**	85.30
κ(%)	58.14	59.97	69.86	71.65	72.74	74.07	73.93	70.48	72.30	72.59	75.94	76.78	**77.31**
time(min) [2]	0 + 0.03	10.11 + 1.05	10.11 + 1.37	10.11 + 1.24	10.11 + 9.24	**1.28** + 13.22	10.11 + 8.37	10.11 + **1.01**	10.11 + 1.32	10.11 + 1.21	10.11+ 8.35	**1.28** + 13.19	10.11 + 8.05

[1] "Sa" denotes **s**patial-**a**daptive-based methods; "Si" denotes **Si**amese-based methods; "-fix" denotes 2D or 3D patches are extracted by **fix**ing a certain sample at the center of neighborhood windows.; "-1" denotes a 2D or 3D residual basic block is used in the ResNet-based methods; "-2" denotes a 2D or 3D ResNeXt basic block is used in the ResNet-based methods. [2] The time before "+" denotes the computation time of patch construction step, while the time after "+" denotes the computation time of classification step.

Table 12. Classification accuracy (%) of various methods [1] for the Zurich data with 10 labeled training samples per class, bold values indicate the best result for a row.

Class	SVM	2D-Based Methods						3D-Based Methods					
		SaCNN	SaResNet-1	SaResNet-2	SaSiResNet-1	SiResNet-2-fix	SaSiResNet-2	SaCNN	SaResNet-1	SaResNet-2	SaSiResNet-1	SiResNet-2-fix	SaSiResNet-2
1	32.15	52.49	45.69	45.49	57.68	**62.17**	57.47	54.98	51.63	49.13	58.12	57.90	53.17
2	66.36	50.45	79.92	84.76	82.83	86.31	84.29	78.93	81.30	83.27	82.88	83.00	**86.71**
3	**90.57**	63.58	88.78	81.09	81.31	74.14	80.80	84.32	82.86	80.57	81.19	82.90	78.50
4	27.79	37.35	64.53	61.41	58.47	52.53	58.31	65.55	65.59	**66.32**	58.45	58.15	64.00
5	70.08	76.54	77.65	78.15	79.12	79.45	79.42	81.42	81.53	77.20	79.81	**83.17**	80.35
6	95.50	**98.79**	94.37	94.99	93.68	96.66	93.99	94.04	94.76	92.80	94.18	95.52	92.97
OA(%)	62.23	53.65	75.09	75.82	76.30	76.80	76.92	75.40	75.89	75.95	76.41	76.91	**77.79**
AA(%)	63.74	63.20	75.16	74.32	75.51	75.21	75.71	76.54	76.28	74.88	75.77	**76.77**	75.95
κ(%)	47.48	40.40	64.68	65.08	66.08	66.72	66.81	65.31	65.77	65.55	66.32	67.40	**67.95**
time(min) [2]	0 + 0.13	**3.22** + 1.01	**3.22** +0.94	**3.22** + 0.86	**3.22** + 3.54	5.62 + 7.94	**3.22** + 3.23	**3.22** + 0.98	**3.22** + 0.92	**3.22** + **0.68**	**3.22**+ 2.91	5.62 + 6.54	**3.22** + 3.12

[1] "Sa" denotes **s**patial-**a**daptive-based methods; "Si" denotes **Si**amese-based methods; "-fix" denotes 2D or 3D patches are extracted by **fix**ing a certain sample at the center of neighborhood windows.; "-1" denotes a 2D or 3D residual basic block is used in the ResNet-based methods; "-2" denotes a 2D or 3D ResNeXt basic block is used in the ResNet-based methods. [2] The time before "+" denotes the computation time of patch construction step, while the time after "+" denotes the computation time of classification step.

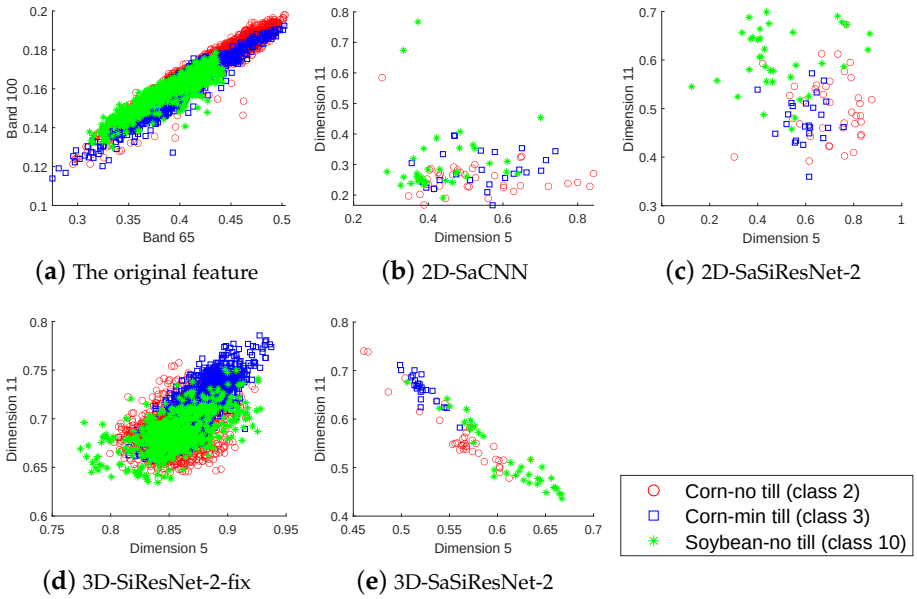


(**a**) SVM (**b**) 2D-SaCNN (**c**) 2D-SaResNet-1 (**d**) 2D-SaResNet-2 (**e**) 2D-SaSiResNet-1

(**f**) 2D-SiResNet-fix (**g**) 2D-SaSiResNet-2 (**h**) 3D-SaCNN (**i**) 3D-SaResNet-1 (**j**) 3D-SaResNet-2

(**k**) 3D-SaSiResNet-1 (**l**) 3D-SiResNet-fix (**m**) 3D-SaSiResNet-2

Figure 12. Classification maps of the Zurich data obtained by (**a**) SVM, (**b**) 2D-SaCNN, (**c**) 2D-SaResNet-1, (**d**) 2D-SaResNet-2, (**e**) 2D-SaSiResNet-1, (**f**) 2D-SiResNet-fix, (**g**) 2D-SaSiResNet-2, (**h**) 3D-SaCNN, (**i**) 3D-SaResNet-1, (**j**) 3D-SaResNet-2, (**k**) 3D-SaSiResNet-1, (**l**) 3D-SiResNet-2-fix and (**m**) 3D-SaSiResNet-2.



(**a**) The original feature (**b**) 2D-SaCNN (**c**) 2D-SaSiResNet-2

(**d**) 3D-SiResNet-2-fix (**e**) 3D-SaSiResNet-2

Figure 13. Scattering map of the two-dimensional features obtained by (**a**) the original data, (**b**) 2D-SaCNN, (**c**) 2D-SaSiResNet-2, (**d**) 3D-SiResNet-2-fix, and (**e**) 3D-SaSiResNet-2 for the Indian Pines data. (**a**,**d**) represent the features of each pixel while (**b**,**c**,**e**) are the features of each patch; therefore, the number of features in object-based methods (i.e., (**b**,**c**,**e**)) are much less than that in the original data (i.e., (**a**)) and the pixel-based method (i.e., (**d**)).

3.4. Parameter Analysis

In this section, the influence of four important parameters, i.e., the number of hidden layers in SAE for band reduction of the hyperspectral image, the number of training samples, the number of superpixels, and the number of training epochs on the performance of the proposed method, is discussed. Figures 14–17 show the OA of the proposed method when the four parameters vary.

As shown in Figure 14, we set the number of hidden layers in SAE as 1 (the number of the unit is set to 5), 3 (the number of units is set to 100, 25 and 5, respectively), 5 (the number of units is set to 100, 50, 25, 10 and 5, respectively), 7 (the number of units is set to 150, 100, 75, 50, 25, 10, and 5, respectively), and 9 (the number of units is set to 175, 150, 125, 100, 75, 50, 25, 10, and 5, respectively), and the classification accuracy varies with the different number of hidden layers. It can be observed that the OA is relatively low when the number of hidden layers is 1 or 3. When the number of hidden layers is equal to or larger than 5, the classification performance is satisfactory. In this paper, we set the number of hidden layers to 5 since more hidden layers will increase the complexity of the model.

Figure 15 depicts the OA of different methods with various numbers of training samples for the Indian Pines data. The number of training samples in classes 1, 7, and 9 is constantly set to 10 in different cases. It can be seen from Figure 15 that the classification performance of all considered methods yields stable improvement with the increase of the training set size. For instance, the OA of 2D-SaCNN is lower than 60% in case 10 training samples per class are selected and is increased to more than 70% with 50 training samples per class. This stresses again the importance of training samples for multiband images classification. Moreover, it can be concluded from Figure 15 that the 3D-SaSiResNet has better classification performance than the Siamese convolutional neural network (S-CNN) proposed in [45]. This is because, when the number of training samples in each class equals 50, the classification accuracy of S-CNN is lower than 90% (see Figure 20 in [45]), while the accuracy of our proposed method is higher than 90% with 16 rather than nine classes of land-covers and using 50 training samples per class.

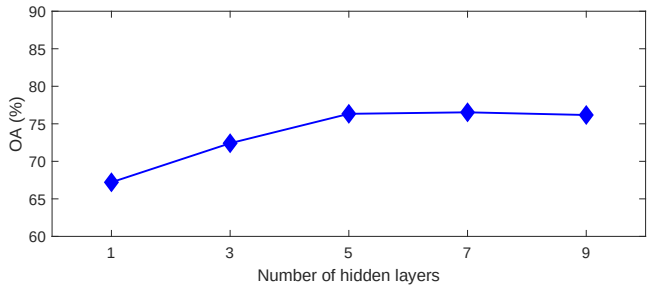


Figure 14. Impact of the number of hidden layers in SAE to the performance of OA for the Indian Pines data.

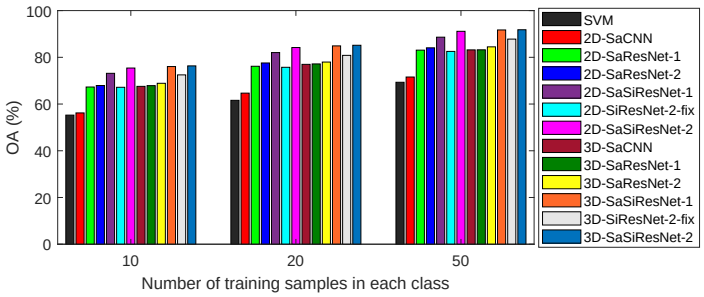


Figure 15. Overall accuracy (%) of different methods with various number of training samples for the Indian Pines data. The number of training samples in classes 1, 7, and 9 is constantly set to 10 in different cases.

Figure 16 plots the OA against the number of superpixels for the Indian Pines data. The number of superpixels is varied within the range from 290 to 2896. We can observe that a dramatically upward tendency appeared with an initial increase of superpixel number, and the classification accuracy decreases in case the number of superpixels is over a certain value (e.g., 1057). Based on the above analysis, too small and too large number of superpixels may be harmful to the classification performance, and it is better to utilize suitable superpixel numbers according to the object characteristics' spatial structure of the remote sensing data.

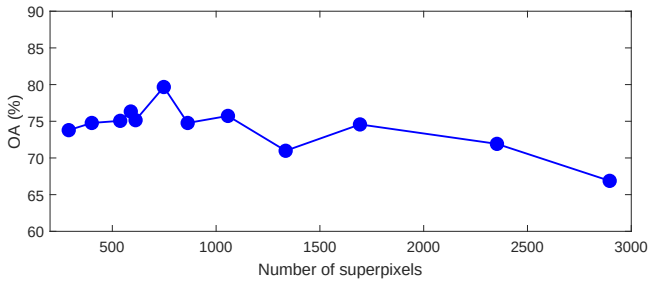


Figure 16. Impact of the number of superpixels on the performance of OA in the proposed 3D-SaSiResNet method for the Indian Pines data.

Furthermore, the impact of the number of training epochs is depicted in Figure 17. It can be seen that the OA rapidly increases in the first couple of epochs and then improves slowly as the number of epochs increases. Finally, the OA trends to a certain stable value with the increased number of epochs. As shown in Figure 17, it is suggested that the 3D-SaSiResNet-2 is trained more than 30 epochs to obtain stable and effective classification performance.

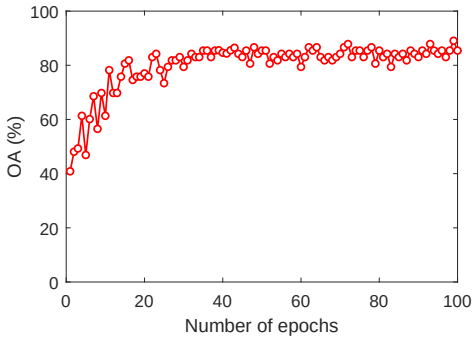


Figure 17. Impact of the number of training epochs to the performance of OA in the proposed 3D-SaSiResNet-2 method for the Indian Pines data.

3.5. Discussion

In this paper, the three datasets are benchmark and widely-used in remote sensing classification. In the experiments, the Indian Pines data gain more improvement in classification accuracy than the other two datasets. The reason why Indian Pines data achieve more discriminative results than the other two datasets is that the spatial resolution of Indian Pines data (i.e., 20 m) is much lower than that of the Pavia (i.e., 1.3 m) and Zurich datasets (i.e., 0.61 m). Therefore, in the Indian Pines data, the region represented by the same number of pixels may contain more different types of land-covers than the other two datasets. It is noteworthy that the proposed 3D-SaSiResNet can better distinguish the boundaries of different land-covers, and our spatial-adaptive-based method has a higher advantage in the classification of Indian Pines data. We perform statistical significance analysis to further confirm the effectiveness of the proposed method. The statistical difference between the proposed 3D-SaSiResNet-2

and competing methods is studied by the McNemar's test, which is achieved by the standardized normal test statistic

$$Z = \frac{f_{12} - f_{21}}{\sqrt{f_{12} + f_{21}}}, \tag{11}$$

where $f_{ij}, (i, j = 1, 2)$ denotes the number of samples correctly classified by the classifier i but incorrectly classified by the classifier j and Z is the pairwise statistical significance of the difference between the ith and jth classifiers. If $|Z| > 1.96$, the classification difference of the two classifiers is considered as statistically significant at the 5% level of significance. It is shown in Table 13 that the 3D-SaSiResNet-2 is statistically superior ($|Z| > 1.96$) to all of the other methods except the 3D-SaSiResNet-1 ($|Z| < 1.96$) for the Indian Pines data. Moreover, Figure 13 intuitively compares the separability of the features learned by different methods according to choosing three difficult-to-separate classes for the Indian Pines data. It is observed that the features obtained by the proposed method are easier to distinguish than other methods.

In the 3D-SaSiResNet, the nearest neighbor classifier is used to classify the learned features of testing samples. The nearest neighbor classifier can be replaced by other sophisticated classifies (e.g., k-nearest neighbor (KNN), SVM or Softmax layer), and it can be treated as a special case of KNN with $k = 1$. It is worth noting that, in case the training set is quite small, the nearest neighbor classifier provides better classification performance than other sophisticated classifiers, while KNN, SVM, or the Softmax layer has higher accuracy than the nearest neighbor classifier if more training samples are available.

The classification results shown in Table 10–12 seems a little bit low. For instance, the highest OA in Table 10 is lower than 80%. In addition, there is a lagging between OA and κ. The reason for the above phenomena is that we only choose a small training set (i.e., training samples per class are no more than 10) in the experiment. The number of training samples is essential to the classification accuracies of various methods. The classification performance of a certain method with a small training set is more likely inferior to the situation with a big training set. As shown in Figure 15, the OA of 3D-SaSiResNet-2 is higher than 90% with 50 training samples per class (except the classes 1, 7, and 9). Moreover, the gap between OA and Kappa will decrease with the increase of training samples.

Table 13. McNemar's test between 3D-SaSiResNet-2 and other methods.

Methods	*Z*		
	Indian Pines Data	**University of Pavia Data**	**Zurich Data**
3D-SaSiResNet-2 vs. SVM	51.35	78.65	194.55
3D-SaSiResNet-2 vs. 2D-SaCNN	49.05	74.38	283.45
3D-SaSiResNet-2 vs. 2D-SaResNet-1	21.96	28.61	29.12
3D-SaSiResNet-2 vs. 2D-SaResNet-2	20.40	20.87	20.14
3D-SaSiResNet-2 vs. 2D-SaSiResNet-1	7.65	19.64	12.97
3D-SaSiResNet-2 vs. 2D-SiResNet-2-fix	22.25	12.38	6.45
3D-SaSiResNet-2 vs. 2D-SaSiResNet-2	2.13	15.31	4.90
3D-SaSiResNet-2 vs. 3D-SaCNN	21.28	29.51	24.52
3D-SaSiResNet-2 vs. 3D-SaResNet-1	20.01	21.54	18.66
3D-SaSiResNet-2 vs. 3D-SaResNet-2	19.35	18.72	17.42
3D-SaSiResNet-2 vs. 3D-SaSiResNet-1	1.19	7.12	10.54
3D-SaSiResNet-2 vs. 3D-SiResNet-2-fix	7.89	2.94	5.51

4. Conclusions

In this paper, we have proposed a deep network architecture, namely 3D-SaSiResNet, for classification of the multiband images data. The underlying idea of 3D-SaSiResNet is to explore the spatial-adaptive and discriminative features that make the samples from the same class close and those belonging to different classes separated. Based on this concept, 3D spatial-adaptive patches are constructed by superpixel segmentation. In contrast to previous approaches that fix the samples at

the center, the patches generated by superpixels do not force the samples at the center but flexibly make the neighbors within the same patch be more likely to come from the same class. Moreover, the network has a Siamese architecture composed of two 3D-ResNet-based subnetworks, which are pairwise trained to increase the separability of different classes. The unlabeled testing samples can be fed into the trained network and classified by a simple nearest neighbor classifier. Experiments on three multiband images datasets acquired by different sensors have demonstrated that the proposed 3D-SaSiResNet method outperforms the state-of-the-art techniques in terms of visual quality and recognition performance. Comparisons show that the 3D-SaSiResNet is especially effective in case limited samples are available for training. The main disadvantage of 3D-SaSiResNet is that the running time of the 3D spatial adaptive patch construction step is much longer than that of the 2D-SiResNet-2-fix and 3D-SiResNet-2-fix. In future works, we will try to adaptively determine the optimal number of superpixels for different datasets and improve the proposed method by deformable convolutions. Classifying unbalanced samples, especially the number of samples in some categories that are particularly small, is one of our future activities. Moreover, it would be a great interest to improve the computational efficiency of the Siamese architecture by pruning and quantization.

Author Contributions: All coauthors made significant contributions to the manuscript. Z.H. designed the research framework, analyzed the results, and wrote the manuscript. D.H. assisted in the prepared work and validation work. Moreover, all coauthors contributed to the editing and review of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Key R&D Program of China, grant numbers 2018YFB0505500 and 2018YFB0505503; the Guangdong Basic and Applied Basic Research Foundation, grant number 2019A1515011877; the Guangzhou Science and Technology Planning Project, grant number 202002030240; the Fundamental Research Funds for the Central Universities, grant number 19lgzd10; the National Natural Science Foundation of China, grant numbers 41501368 and 4153117; the Southern Marine Science and Engineering Guangdong Laboratory (Zhuhai), grant number 99147-42080011; the 2018 Key Research Platforms and Research Projects of Ordinary Universities in GuangDong Province, grant number 2018KQNCX360.

Acknowledgments: The authors would like to take this opportunity to thank the Editors and the Anonymous Reviewers for their detailed comments and suggestions, which greatly helped us to improve the clarity and presentation of our manuscript.

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

References

1. Sun, W.; Jiang, M.; Li, W.; Liu, Y. A symmetric sparse representation based band selection method for hyperspectral imagery classification. *Remote Sens.* **2016**, *8*, 238. [CrossRef]
2. Li, J.; Liu, Z. Multispectral transforms using convolution neural networks for remote sensing multispectral image compression. *Remote Sens.* **2019**, *11*, 759. [CrossRef]
3. Feng, J.; Feng, X.; Chen, J.; Cao, X.; Zhang, X.; Jiao, L.; Yu, T. Generative adversarial networks based on collaborative learning and attention mechanism for hyperspectral image classification. *Remote Sens.* **2020**, *12*, 1149. [CrossRef]
4. Nezami, S.; Khoramshahi, E.; Nevalainen, O.; Pölönen, I.; Honkavaara, E. Tree species classification of drone hyperspectral and rgb imagery with deep learning convolutional neural networks. *Remote Sens.* **2020**, *12*, 1070. [CrossRef]
5. Padró, J.C.; Muñoz, F.J.; Planas, J.; Pons, X. Comparison of four UAV georeferencing methods for environmental monitoring purposes focusing on the combined use with airborne and satellite remote sensing platforms. *Int. J. Appl. Earth Obs. Geoinf.* **2019**, *75*, 130–140. doi:10.1016/j.jag.2018.10.018. [CrossRef]
6. Zheng, Z.; Du, S.; Wang, Y.C.; Wang, Q. Mining the regularity of landscape-structure heterogeneity to improve urban land-cover mapping. *Remote Sens. Environ.* **2018**, *214*, 14–32. doi:10.1016/j.rse.2018.05.019. [CrossRef]
7. Olteanu-Raimond, A.M.; See, L.; Schultz, M.; Foody, G.; Riffler, M.; Gasber, T.; Jolivet, L.; le Bris, A.; Meneroux, Y.; Liu, L.; et al. Use of automated change detection and vgi sources for identifying and validating urban land use change. *Remote Sens.* **2020**, *12*, 1186. [CrossRef]

8. Yao, X.; Zhao, C. Hyperspectral anomaly detection based on the bilateral filter. *Infrared Phys. Technol.* **2018**, *92*, 144–153. doi:10.1016/j.infrared.2018.05.028. [CrossRef]
9. Fauvel, M.; Tarabalka, Y.; Benediktsson, J.A.; Chanussot, J.; Tilton, J.C. Advances in spectral-spatial classification of hyperspectral images. *Proc. IEEE* **2013**, *101*, 652–675. doi:10.1109/jproc.2012.2197589. [CrossRef]
10. Lu, X.; Zheng, X.; Yuan, Y. Remote sensing scene classification by unsupervised representation learning. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 5148–5157. doi:10.1109/tgrs.2017.2702596. [CrossRef]
11. Li, Z.; Tang, X.; Li, W.; Wang, C.; Liu, C.; He, J. A two-stage deep domain adaptation method for hyperspectral image classification. *Remote Sens.* **2020**, *12*, 1054. [CrossRef]
12. Cortes, C.; Vapnik, V. Support-vector networks. *Machine learning* **1995**, *20*, 273–297. [CrossRef]
13. Chang, C.C.; Lin, C.J. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.* **2011**, *2*, 27:1–27:27. Available online: http://www.csie.ntu.edu.tw/~cjlin/libsvm (accessed on 2 May 2020).
14. Paoletti, M.E.; Haut, J.M.; Tao, X.; Miguel, J.P.; Plaza, A. A new GPU implementation of support vector machines for fast hyperspectral image classification. *Remote Sens.* **2020**, *12*, 1257. [CrossRef]
15. Benediktsson, J.A.; Palmason, J.A.; Sveinsson, J.R. Classification of hyperspectral data from urban areas based on extended morphological profiles. *IEEE Trans. Geosci. Remote Sens.* **2005**, *43*, 480–491. doi:10.1109/TGRS.2004.842478. [CrossRef]
16. Mura, M.D.; Villa, A.; Benediktsson, J.A.; Chanussot, J.; Bruzzone, L. Classification of Hyperspectral Images by Using Extended Morphological Attribute Profiles and Independent Component Analysis. *IEEE Geosci. Remote Sens. Lett.* **2011**, *8*, 542–546. doi:10.1109/LGRS.2010.2091253. [CrossRef]
17. Tsai, F.; Lai, J.S. Feature extraction of hyperspectral image cubes using three-dimensional gray-level cooccurrence. *IEEE Trans. Geosci. Remote Sens.* **2013**, *51*, 3504–3513. doi:10.1109/tgrs.2012.2223704. [CrossRef]
18. Jia, S.; Shen, L.; Li, Q. Gabor feature-based collaborative representation for hyperspectral imagery classification. *IEEE Trans. Geosci. Remote Sens.* **2015**, *53*, 1118–1129. doi:10.1109/tgrs.2014.2334608. [CrossRef]
19. Yin, J.; Gao, C.; Jia, X. Wavelet Packet Analysis and Gray Model for Feature Extraction of Hyperspectral Data. *IEEE Geosci. Remote Sens. Lett.* **2013**, *10*, 682–686. doi:10.1109/LGRS.2012.2218569. [CrossRef]
20. Achanta, R.; Shaji, A.; Smith, K.; Lucchi, A.; Fua, P.; Süsstrunk, S. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 2274–2282. doi:10.1109/tpami.2012.120. [CrossRef]
21. Liu, T.; Gu, Y.; Chanussot, J.; Mura, M.D. Multimorphological Superpixel Model for Hyperspectral Image Classification. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 6950–6963. doi:10.1109/tgrs.2017.2737037. [CrossRef]
22. Zhang, H.; Li, J.; Huang, Y.; Zhang, L. A nonlocal weighted joint sparse representation classification method for hyperspectral imagery. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2014**, *7*, 2056–2065. doi:10.1109/jstars.2013.2264720. [CrossRef]
23. He, Z.; Wang, Q.; Shen, Y.; Sun, M. Kernel sparse multitask learning for hyperspectral image classification with empirical mode decomposition and morphological wavelet-based features. *IEEE Trans. Geosci. Remote Sens.* **2014**, *52*, 5150–5163. doi:10.1109/TGRS.2013.2287022. [CrossRef]
24. Tarabalka, Y.; Fauvel, M.; Chanussot, J.; Benediktsson, J.A. SVM- and MRF-based method for accurate classification of hyperspectral images. *IEEE Geosci. Remote Sens. Lett.* **2010**, *7*, 736–740. doi:10.1109/lgrs.2010.2047711. [CrossRef]
25. Zhang, L.; Zhang, L.; Tao, D.; Huang, X. Tensor discriminative locality alignment for hyperspectral image spectral-spatial feature extraction. *IEEE Trans. Geosci. Remote Sens.* **2013**, *51*, 242–256. doi:10.1109/tgrs.2012.2197860. [CrossRef]
26. He, Z.; Li, J.; Liu, L. Tensor block-sparsity based representation for spectral-spatial hyperspectral image classification. *Remote Sens.* **2016**, *8*, 636. doi:10.3390/rs8080636. [CrossRef]
27. Zhong, Z.; Fan, B.; Duan, J.; Wang, L.; Ding, K.; Xiang, S.; Pan, C. Discriminant tensor spectral-spatial feature extraction for hyperspectral image classification. *IEEE Geosci. Remote Sens. Lett.* **2015**, *12*, 1028–1032. doi:10.1109/lgrs.2014.2375188. [CrossRef]
28. 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. Mag.* **2016**, *4*, 22–40. [CrossRef]

29. 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]
30. Wang, S.; Quan, D.; Liang, X.; Ning, M.; Guo, Y.; Jiao, L. A deep learning framework for remote sensing image registration. *ISPRS J. Photogramm. Remote Sens.* **2018**, *145*, 148–164. doi:10.1016/j.isprsjprs.2017.12.012. [CrossRef]
31. Dong, G.; Liao, G.; Liu, H.; Kuang, G. A review of the autoencoder and its variants: A comparative perspective from target recognition in synthetic-aperture radar images. *IEEE Geosci. Remote Sens. Mag.* **2018**, *6*, 44–68. doi:10.1109/mgrs.2018.2853555. [CrossRef]
32. 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]
33. Chen, Y.; Zhao, X.; Jia, X. Spectral-spatial classification of hyperspectral data based on deep belief network. *IEEE J. Sel. Topics Appl. Earth Obs. Remote Sens.* **2015**, *8*, 2381–2392. doi:10.1109/jstars.2015.2388577. [CrossRef]
34. 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, 26 June–1 July 2016; pp. 770–778.
35. He, Z.; Liu, H.; Wang, Y.; Hu, J. Generative adversarial networks-based semi-supervised learning for hyperspectral image classification. *Remote Sens.* **2017**, *9*, 1042. doi:10.3390/rs9101042. [CrossRef]
36. Hu, W.; Huang, Y.; Wei, L.; Zhang, F.; Li, H. Deep convolutional neural networks for hyperspectral image classification. *J. Sens.* **2015**, *2015*, 1–12. doi:10.1155/2015/258619. [CrossRef]
37. Yue, J.; Zhao, W.; Mao, S.; Liu, H. Spectral-spatial classification of hyperspectral images using deep convolutional neural networks. *Remote Sens. Lett.* **2015**, *6*, 468–477. doi:10.1080/2150704x.2015.1047045. [CrossRef]
38. Cheng, G.; Yang, C.; Yao, X.; Guo, L.; Han, J. When deep learning meets metric learning: remote sensing image scene classification via learning discriminative CNNs. *IEEE Trans. Geosci. Remote Sens.* **2018**, *56*, 2811–2821. doi:10.1109/tgrs.2017.2783902. [CrossRef]
39. Li, Y.; Zhang, H.; Shen, Q. Spectral-spatial classification of hyperspectral imagery with 3D convolutional neural network. *Remote Sens.* **2017**, *9*, 67. doi:10.3390/rs9010067. [CrossRef]
40. 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. doi:10.1109/tgrs.2018.2818945. [CrossRef]
41. 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.* **2018**, *56*, 847–858. doi:10.1109/tgrs.2017.2755542. [CrossRef]
42. Koch Gregory, R.Z.; Salakhutdinov, R. Siamese neural networks for one-shot image recognition. In Proceedings of the ICML Deep Learning Workshop, Lille, France, 6–11 July 2015; Volume 2.
43. Liu, B.; Yu, X.; Yu, A.; Zhang, P.; Wan, G.; Wang, R. Deep Few-Shot Learning for Hyperspectral Image Classification. *IEEE Trans. Geosci. Remote Sens.* **2018**, *57*, 2290–2304. [CrossRef]
44. Liu, B.; Yu, X.; Yu, A.; Wan, G. Deep convolutional recurrent neural network with transfer learning for hyperspectral image classification. *J. Appl. Remote Sens.* **2018**, *12*, 026028. [CrossRef]
45. Liu, B.; Yu, X.; Zhang, P.; Yu, A.; Fu, Q.; Wei, X. Supervised deep feature extraction for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **2017**, *56*, 1909–1921. [CrossRef]
46. Zhao, S.; Li, W.; Du, Q.; Ran, Q. Hyperspectral classification based on siamese neural network using spectral-spatial feature. In Proceedings of the IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium, Valencia, Spain, 22–27 July 2018; pp. 2567–2570.
47. Hinton, G.E.; Zemel, R.S. Autoencoders, minimum description length and helmholtz free energy. In *Advances in Neural Information Processing Systems 6*; Cowan, J.D., Tesauro, G., Alspector, J., Eds.; Morgan-Kaufmann: Burlington, MA, USA, 1994; pp. 3–10.
48. Bengio, Y.; Lamblin, P.; Popovici, D.; Larochelle, H. Greedy layer-wise training of deep networks. In Proceedings of the 19th International Conference on Neural Information Processing Systems, Doha, Qatar, 12–15 November 2006; MIT Press: Cambridge, MA, USA, 2006; pp. 153–160.
49. Van Der Maaten, L.; Postma, E.; Van den Herik, J. Dimensionality reduction: a comparative review. *J. Mach. Learn. Res.* **2009**, *10*, 66–71.

50. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv* **2015**, arXiv:1502.03167.
51. Nair, V.; Hinton, G.E. Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), Haifa, Israel, 21–24 June 2010; pp. 807–814.
52. Boureau, Y.L.; Ponce, J.; LeCun, Y. A theoretical analysis of feature pooling in visual recognition. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), Haifa, Israel, 21–24 June 2010; pp. 111–118.
53. Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; He, K. Aggregated residual transformations for deep neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1492–1500.
54. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
55. Volpi, M.; Ferrari, V. Semantic segmentation of urban scenes by learning local class interactions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 7–12 June 2015; pp. 1–9.
56. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.