remote sensing

MDPI

Article

A Lightweight Spectral–Spatial Feature Extraction and Fusion Network for Hyperspectral Image Classification

Linlin Chen [1,2], Zhihui Wei [1,*] and Yang Xu [1]

[1] School of Computer Science and Engineering, Nanjing University of Science & Technology, Nanjing 210094, China; chenlinlin606@njust.edu.cn (L.C.); xuyangth90@njust.edu.cn (Y.X.)

[2] Zijin College, Nanjing University of Science & Technology, Nanjing 210023, China

* Correspondence: gswei@njust.edu.cn; Tel.: +86-25-84318108

Received: 1 April 2020; Accepted: 24 April 2020; Published: 28 April 2020

Abstract: Hyperspectral image (HSI) classification accuracy has been greatly improved by employing deep learning. The current research mainly focuses on how to build a deep network to improve the accuracy. However, these networks tend to be more complex and have more parameters, which makes the model difficult to train and easy to overfit. Therefore, we present a lightweight deep convolutional neural network (CNN) model called S2FEF-CNN. In this model, three S2FEF blocks are used for the joint spectral–spatial features extraction. Each S2FEF block uses 1D spectral convolution to extract spectral features and 2D spatial convolution to extract spatial features, respectively, and then fuses spectral and spatial features by multiplication. Instead of using the full connected layer, two pooling layers follow three blocks for dimension reduction, which further reduces the training parameters. We compared our method with some state-of-the-art HSI classification methods based on deep network on three commonly used hyperspectral datasets. The results show that our network can achieve a comparable classification accuracy with significantly reduced parameters compared to the above deep networks, which reflects its potential advantages in HSI classification.

Keywords: hyperspectral image classification; spectral–spatial feature extraction and fusion; deep learning; convolutional neural network

1. Introduction

Hyperspectral remote sensing technology is a focus in the remote sensing field, which has been applied for crop management, image segmentation, object recognition, etc. [1–4]. Hyperspectral image classification often plays the most important role of these applications. In HSI, each pixel is always considered as a high-dimensional vector. So, the classification of HSI is essentially to predict a specific category for each pixel according to its characteristics [5].

According to the way in which hyperspectral image features are acquired, we classify the hyperspectral image (HSI) classification methods into two categories: one is the method that extracts the HSI features manually, while the other is the method that extracts the HSI features automatically.

The traditional HSI classification methods belong to the first category, most of which analyze the HSIs and extract their shallow features for classification. The most prominent feature of HSI is the rich spectral information. Early researches concentrated on acquiring accurate and efficient spectrum characteristics [6–10]. For example, the authors of [6] and [7] used spectral angle or spectral divergence for pixel matching. In addition, the authors of [8–10] used another kind of method based on statistics, completed the classification by learning from the labeled samples. In [11], a more accurate feature extraction based on Principal Components Analysis (PCA) was used. However, the HSI data includes not only spectral features of each pixel but also the spatial relationship between these pixels. Only using the spectral information for classification leads to low accuracy. Therefore, current research on HSI classification mostly uses the spectral–spatial features for classification [12–19]. In [16], a controlled random sampling strategy was used to get a more accurate training set and testing set. In [17], the spectrum was first partitioned into several groups and then band-specific spectral–spatial features were extracted by a convolutional neural network (CNN). In [18], the Gabor features were stacked through Gabor filters with spectral features for classification. All of this research has proven that using spectral–spatial joint features for HSI classification can effectively improve classification accuracy.

CNN is a representative deep learning model [20–23]. The task of HSI classification, according to the different features processed by CNN, falls into three categories. The first category is 1D-CNN based on spectral features only. In 2015, Wei et al. [24] applied a five-layer CNN to HSI classification and proposed the 1D-CNN network to extract spectral features. Li et al. [14] proposed the idea of pixel pairs, learning the spectral features of a pair of pixels, and predicted the final classification results by voting. Yue [25] used preprocessing before feature extraction. Mei et al. [26] preprocessed each pixel with the mean and standard deviation from its neighborhood. The second category is a spatial feature-based method called 2D-CNN. The authors of [15] introduced L2 regularization and virtual samples. In [27], the Bayesian method was introduced. In [28], spatial features were expressed by sparse representation. The authors of [29] proposed a classification network on specific scenes. The last category is the spectral–spatial feature based method. In this case, there are two different ways of feature processing. One is 3D-CNN. Three-dimensional convolution was first used for video processing, and is now widely used in HSI classification [30–34]. The other is hybrid CNN. Various different applications have been proposed [35–38]. For example, different hybrid ways of using 1D-CNN and 2D-CNN were presented in [35–37]. While the authors of [38] proposed a hybrid network of 3D-CNN and 2D-CNN.

In addition to CNN, other different deep models have been proposed and applied in HSI classification, such as stacked autoencoder (SAE) [39], deep belief network (DBN) [40], deep restricted Boltzmann machine (DRBM) [41], capsule networks [42], dense block [43], and others.

Current research of HSI classification based on deep learning mainly focuses on how to build deep networks to improve accuracy. However, the more complex are these networks, the more training parameters are there. For example, there are about 360,000 training parameters in the classification network proposed in [31]. In [44], the proposed 3D-1D hybrid CNN method used a maximum of 61,949 parameters. The network in [38], a 3D–2D hybrid CNN, used 5,122,176 parameters. Using so many training parameters makes the network difficult to train and easy to overfit.

In this paper, we present a lightweight spectral–spatial feature extraction and fusion convolutional neural network (S2FEF-CNN). In this model, three S2FEF blocks are concatenated to provide the joint spectral–spatial features. Each S2FEF block uses 1D and 2D convolution for spectral and spatial feature extraction, respectively, and then fuses the spectral and spatial features by multiplication. Pooling layers are used for dimension reduction and finally prediction of the classification results.

The main contributions of our work are as follows: 1) it proposes a lightweight end-to-end network model for hyperspectral image classification. In comparison with several state-of-the-art HSI classification methods based on deep networks, most of the time, our network can achieve comparable classification accuracy using no more than 5% of the parameters of the above deep networks, 2) it proposes a dual-channel feature extraction and feature fusion method of HSI classification, 3) without using the fully connected (FC) layer and PCA, it also greatly reduces the network parameters.

The rest of this paper is organized as follows. Section 2 is a brief introduction of various related CNN frameworks. The proposed S2FEF-CNN is illustrated in detail in Section 3. Experimental results and analysis are given in Section 4. Section 5 presents conclusions.

2. Related Work

One-dimensional-CNN uses spectral features for classification, in which the data input is usually a single pixel. The feature map extracted by 1D convolution reflects the characteristics of the spectrum. A typical 1D convolution and 1D-CNN classification framework [24] are shown in Figure 1.

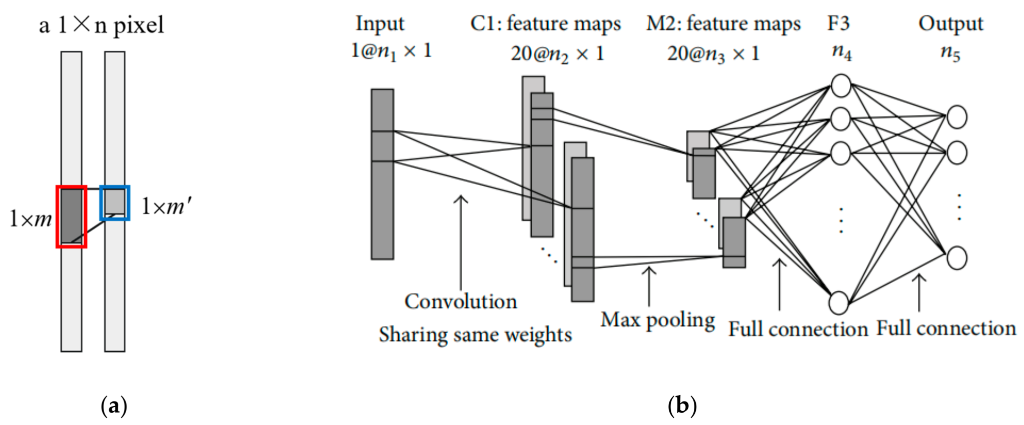


Figure 1. (**a**) A typical 1D convolution; (**b**) 1D-CNN classification framework in [24]. In this framework, there are several layers: 1D convolutional layer, pooling layer, fully-connected (FC) layer, and an output layer. CNN: convolutional neural network.

Two-dimensional-CNN uses spatial features for classification. With spatial features extraction, there is redundancy between spectrum bands. Therefore, this framework is usually combined with PCA to reduce the spectral dimension before spatial feature extraction [18,36]. First, the HSIs are preprocessed by PCA for dimension reduction. Then, diverse spatial features are extracted by 2D convolution kernel. A typical 2D convolution and the classification framework of 2D-CNN [15] are shown in Figure 2.

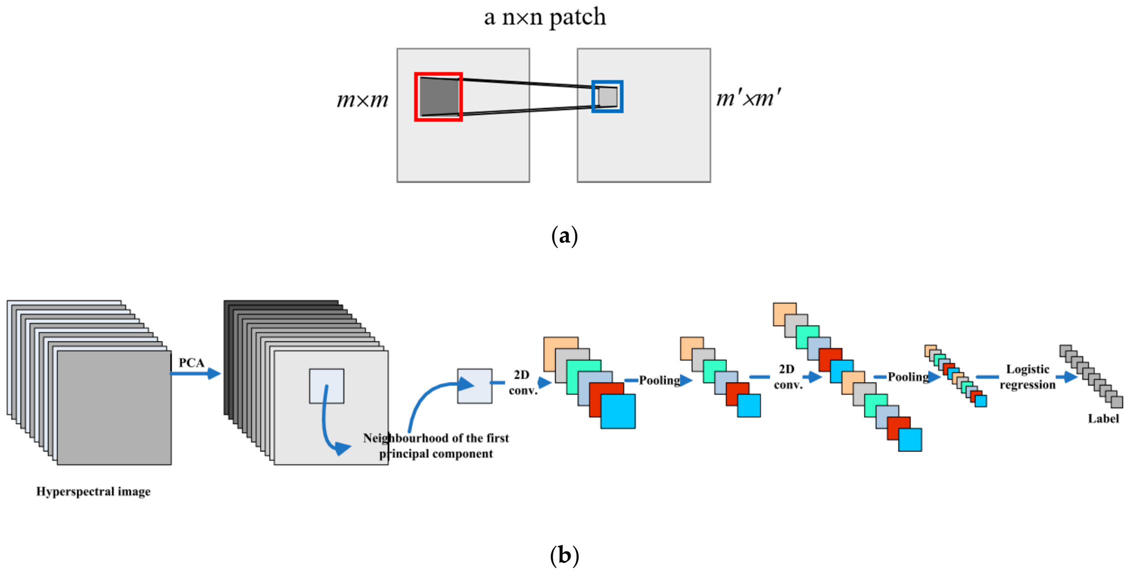
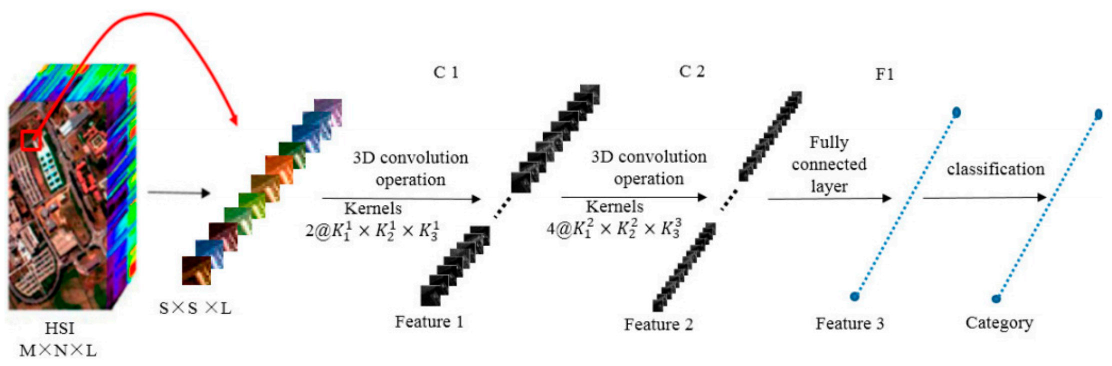
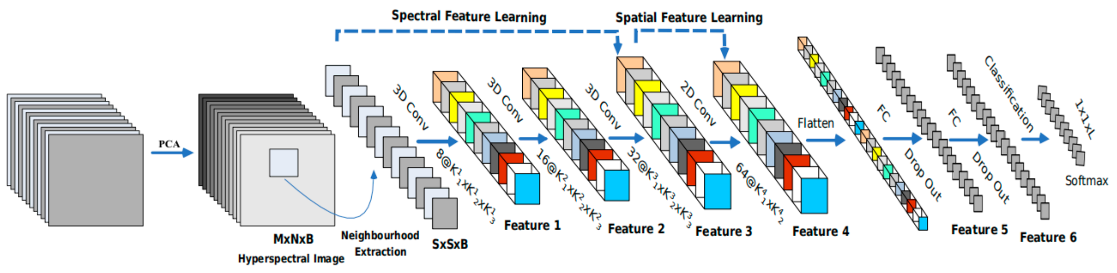


Figure 2. (**a**) A typical 2D convolution; (**b**) 2D-CNN classification framework in [15]. After PCA for dimension reduction, the framework contains two 2D convolutional layers, two pooling layers, and an output layer.

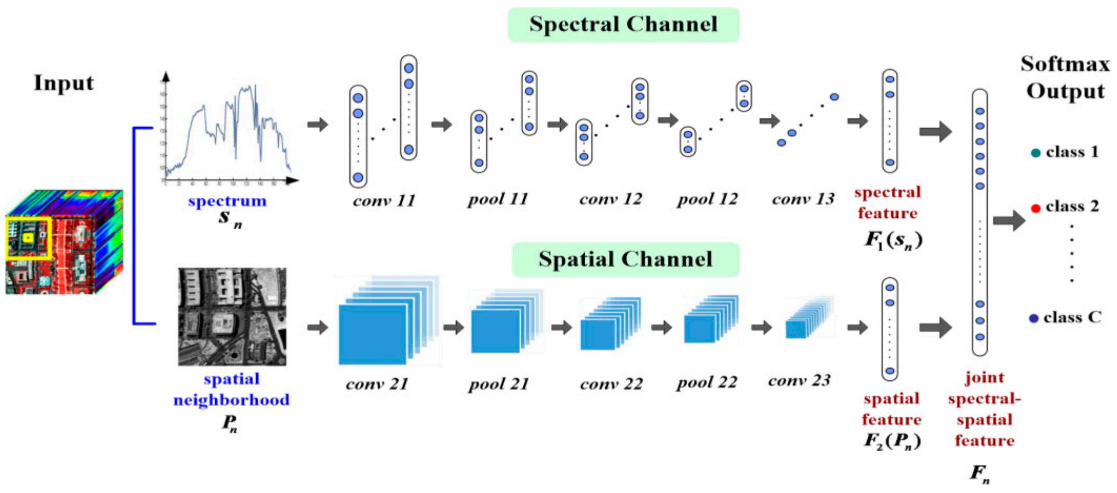
As described above, in the current HSI classification research, there are two kinds of spectral–spatial feature-based CNN classification methods: 3D-CNN and hybrid CNN. The 3D-CNN method extracts features in the spectral and spatial dimensions simultaneously through 3D convolution. However the hybrid CNN selects some of the above mentioned types of CNNs and fuse them in a sequential way or in a multi-channel way. Figure 3 shows the schematic diagram of the above two classes.



(**a**)



(**b**)



(**c**)

Figure 3. Two categories of spectral–spatial feature-based CNN classification methods. (**a**) 3D-CNN classification framework [30]; (**b**) a sequential 3D–2D hybrid CNN framework [38]; (**c**) a multi-channel 1D–2D hybrid CNN framework [37].

In summary, 1D-CNN makes full use of the spectral features of the HSI, but lacks the spatial features. Two-dimensional-CNN has to deal with high spectral dimension problems before extracting spatial features. PCA is often used for reducing spectral dimension. It simplifies the process but brings information loss and destruction of the internal structure of the data. Three-dimensional-CNN can make the process of feature extraction and fusion easier, but the prominent problem is that the model suffers from heavy network parameters. If only the number of network parameters is taken into account, the dual-channel hybrid CNN method in Figure 3c seems to be more suitable and effective.

3. Proposed Methodology

In this section, we first illustrate the structure of the elementary block of our model, and then show in detail how the block extracts and fuses the features. Finally, we elaborate on the architecture of the S2FEF-CNN.

3.1. S2FEF Block

Details of the basic S2FEF block are demonstrated in Figure 4.

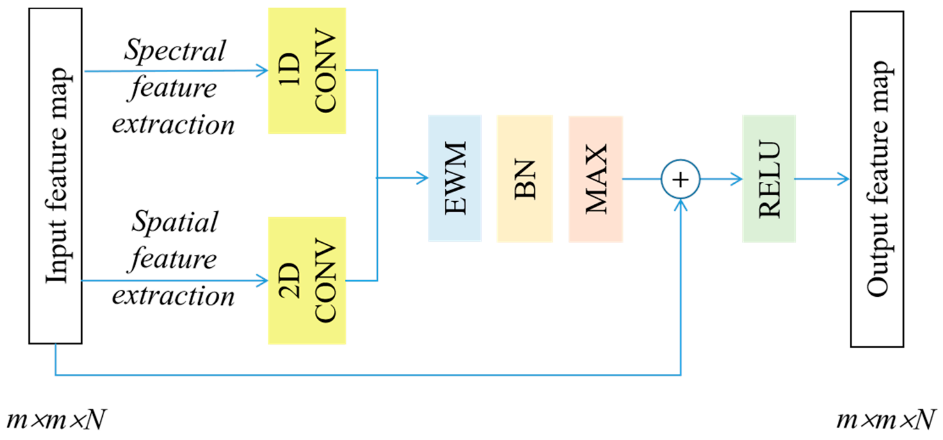


Figure 4. Details of the proposed S2FEF block. EWM is element-wise multiplication, and BN is batch- normalization.

The S2FEF block contains two stages: one is for feature extraction and the other is for feature fusion.

In the first stage, spectral and spatial features are extracted by 1D/2D convolutional kernels in spectral and spatial channels, respectively. This step is formulated as follow:

$$f_{e_t}^j(x_i^j) = W_{e_t}^j x_i^j + b_{e_t}^j \tag{1}$$

$$f_{a_t}^j(x_i^j) = W_{a_t}^j x_i^j + b_{a_t}^j \tag{2}$$

where x_i^j denotes the input HSI data, i = 1, . . . ,I (I is the input number), $W_{e_t}^j$ and $b_{e_t}^j$/$W_{a_t}^j$ and $b_{a_t}^j$ are the weights and bias of spectral/spatial kernel t in layer j, respectively, t = 1, . . . , K_k (K_k is the number of kernels), j represents the layer index, f is the features extractor, and subscript e and a represent spectral and spatial, respectively.

In the following stage, $f_{e_t}^j(x_i^j)$ and $f_{a_t}^j(x_i^j)$ are fused in three steps.

(1) Features from two channels are fused by element-wise multiplication.

$$EWM(x_i^j) = \{f_{e_m}^j(x_i^j) \times f_{a_m}^j(x_i^j)\}_{m=1,\ldots,K_k} \tag{3}$$

After the former feature extraction, we directly fuse the spectral features and spatial features by element-wise multiplication (EWM). Compared with feature concatenation, the EWM will not increase the feature dimension but will adjust the spectral features by spatial information to a certain extent.

(2) The maximum element in different feature cubes are selected to produce the final feature cube.

$$Max(x_i^j) = \max(EWM(x_i^j)) \tag{4}$$

(3) The maximal feature cube selected by Equation (5) is added to the original input cube x_i^j to form a more accurate output cube x_i^{j+1}. Finally, rectified linear unit (Relu) is exploited for activation.

$$x_i^{j+1} = \max(0,\ x_i^j + Max(x_i^i)) \tag{5}$$

The above process is summarized in Algorithm 1.

Algorithm 1: Feature Extraction with S2FEF Block

```
Input: A joint spectral–spatial feature map F,
       Spectral/Spatial kernel size $S_{pe}/S_{pa}$ and kernel number k.
Output: A new joint spectral–spatial feature map F′.
1. begin
2. Extract spectral/spatial features $f_{spe}/f_{spa}$ with k spectral/spatial kernel (size $1 \times 1 \times S_{pe}/S_{pa} \times S_{pa} \times 1$).
3. Fuse the spectral and spatial features together by element-wise multiplication ($f_{spe} \times f_{spa}$) to get the joint features $f_{joint}$.
4. Select the max value from the corresponding pixel in $f_{joint}$ to form a special feature F′.
5. Return F′.
6. end
```

3.2. S2FEF-CNN Architecture

The proposed network mainly consists of three steps:

(1) *Step* 1: extracting spectral–spatial joint features by three S2FEF blocks;

(2) *Step* 2: reducing spectral and spatial dimensions of the joint features by two pooling layers;

(3) *Step* 3: determining the pixel label via a softmax layer after flattening the joint features from Step 2 into a vector.

The architecture of the proposed S2FEF-CNN is shown in Figure 5.

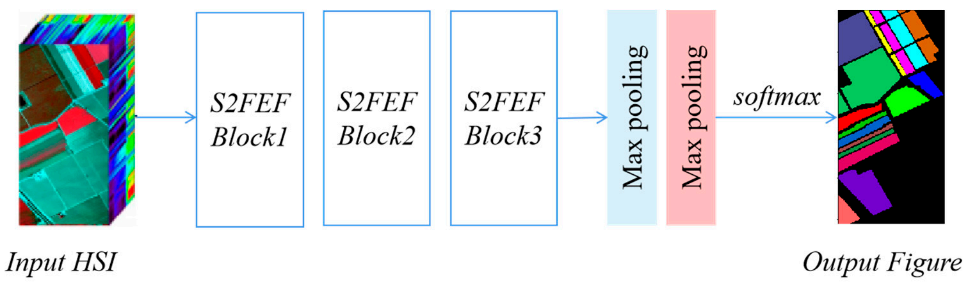


Figure 5. Architecture of the S2FEF-CNN. The two pooling layers are used in the spectral and spatial dimensions, respectively.

Define $x_i \in \mathbb{R}^{m_1 \times m_1 \times N_1}$ (i = 1,2, … ,K) as the input HSI cube and $\hat{y}_i$ as the output label of x_i. The process is defined as follow:

$$\hat{y}_i = \delta_s\ (\delta_p\ (S2FEF\ (x_i))) \tag{6}$$

where *S2FEF* denotes the operator of the proposed spectral–spatial feature extraction and fusion, δ_p denotes two max pooling operators, and δ_s denotes the softmax classification.

Most of the commonly used 2D-CNN classification methods use PCA to preprocess HSIs for dimension reduction, but PCA introduces the problem of spectral information loss. In our architecture, we abandoned PCA and used the full spectrum of HSIs as input, and two pooling layers were used for feature dimension reduction.

CNN-based classification frameworks usually have one or two FC layers to integrate the features before the final classification. However, the FC layer is a heavily weighted layer, which may account for 80% of the parameters of a network. Hence, we also dropped the FC layer in our architecture for network parameter reduction.

Improvements of the architecture make our proposed network light. Experimental results show that even with only a few thousand parameters, our network can achieve comparative classification accuracy as those state-of-the-art deep networks with heavy weight.

The above steps are summarized in Algorithm 2.

Algorithm 2: S2FEF-CNN Classification

```
Input: Hyperspectral image cube size m, S2FEF block number $K_k$.
Output: The class label L of each pixel.
1. begin
2. Create input data set I in which each pixel input cube is size m × m × N (N is the spectrum band number).
3. For each pixel in training set from I.
4. Extract spectral–spatial features by $K_k$ S2FEF blocks.
5. The joint feature is pooled by two max pooling layers after which the feature size is m′ × m′ × N′.
6. Flatten the feature into a vector v.
7. Computing the softmax output L.
8. Return L.
9. end
```

4. Experimental Results

4.1. Datasets

We evaluated our work on three public hyperspectral datasets, Indian Pines (IP), Salinas (SA), and Pavia University (PU), captured by two different sensors: AVIRIS and ROSIS-03. AVIRIS can provide HSI with 224 contiguous spectral bands, covering wavelengths from 0.4 to 2.5 μm and with a spatial resolution of 20 m/pixel, while ROSIS-03 delivers HSI in 115 bands with a spectral coverage ranging from 0.43 to 0.86 μm and with a spatial resolution of 1.3 m/pixel. Indian Pines (IP) and Salinas (SA) are two commonly used datasets by AVIRIS. IP is captured in Northwestern Indiana and is 145 × 145 pixels in size, and Salinas is recorded over Salinas Valley and includes 512 × 217 pixels. They both consist of 16 ground-truth classes. The scene of PU is captured by ROSIS-03 with a size of 610 × 340 pixels and contains nine different classes. In our experiments, we use a corrected version with 200/204 bands for IP and SA, and 103 bands for PU in experiments after removing the noisy bands and a blank strip.

4.2. Parameters Setting

In our network, three S2FEF blocks were used for all the three datasets. In each block, the number and size of convolutional kernels were the same for 1D convolution and 2D convolution. We empirically set the parameters just as the other networks do. The spectral kernel size was 1×3, the spatial kernel size was 3 × 3, and the kernel number was 4.

The input HSI cube size was set differently for each dataset. Unlike some networks that use a small size input, we wanted the original input cube to contain enough spatial information, and then chose a big size input. For the IP dataset, the size was 19 × 19 × N (i.e., m = 19), where N represented the band number. For the PU dataset, the size was 15 × 15 × N, while for the SA dataset, the size was 21 × 21.

We compared our experimental results with four well-performed deep network based HSI classification methods: SAE [39], 1D-CNN [24], 3D-CNN [30], and DC-CNN [36]. First, we compare the number of parameters in Section 4.3, and the results of classification accuracy are described in Sections 4.4–4.7.

4.3. Comparison of Parameter Numbers

In our S2FEF-CNN architecture, the parameters are contained in the S2FEF block and in the final output layer. Detailed analysis results are given in Tables 1 and 2, respectively.

Table 1. Number of parameters in convolutional layers of each S2FEF block.

Layer/Operation	Input	Kernel Size	Kernel Number	Output	Parameters
Spectral channel	$15 \times 15 \times N \times 1$	1×3	4	$15 \times 15 \times N \times 4$	16
Spatial channel	$15 \times 15 \times N \times 1$	3×3	4	$15 \times 15 \times N \times 4$	40
EWM	$15 \times 15 \times N \times 4$	-	-	$15 \times 15 \times N \times 4$	0
Batch normalization	$15 \times 15 \times N \times 4$	-	-	$15 \times 15 \times N \times 4$	8
Relu	$15 \times 15 \times N \times 4$			$15 \times 15 \times N \times 1$	0
Total					64

Table 2. Number of parameters in the S2FEF-CNN architecture for different datasets.

Layer/Operation	Indian Pines	Salinas	Pavia University
S2FEF Block1	64	64	64
S2FEF Block2	64	64	64
S2FEF Block3	64	64	64
Max Pooling layers	0	0	0
Softmax layer	5776	5792	1629
Total	5968	5984	1821

Each S2FEF block has the same number of parameters. For the entire network, a large number of parameters are in the softmax layers, depending on the characteristics of the dataset. For Indian Pines and Salinas datasets, they have more input spectral bands and more ground truth classes, so they have more parameters. Pavia University has fewer bands and fewer classes, so it has fewer parameters.

For SAE, 1D-CNN, 3D-CNN, and DC-CNN, we used the same architecture and the same parameter settings as in their papers. For those settings that are not explicitly given in the paper, we adopted the commonly used values in HSI classification (e.g., the pooling stride was 2).

Table 3 shows the comparison results in detail. Table 4 gives the parameter percentage of S2FEF-CNN compared with other networks.

Table 3. The total parameters of the different networks on different datasets.

Dataset	SAE	1D-CNN	3D-CNN	DC-CNN	S2FEF-CNN
IP	129,856	81,408	199,040	278,552	5968
PU	129,856	61,249	111,129	153,117	1821
SA	123,609	82,216	199,040	278,552	5984

Table 4. The parameter percentage of different networks.

Dataset	S2FEF-CNN/SAE	S2FEF-CNN/1D-CNN	S2FEF-CNN/3D-CNN	S2FEF-CNN/DC-CNN
IP	4.60%	7.33%	3.00%	2.14%
PU	1.40%	2.97%	1.64%	1.19%
SA	4.84%	7.27%	3.01%	2.14%

Obviously, our proposed network works well with the fewest parameters, most of which are no more than 5% of the parameters used by the other deep networks, and the highest percentage is no more than 8%. It seems to be a potentially feasible way to solve the problem of heavy weights while training a deep network.

4.4. Results of the Indian Pines Dataset

This dataset is a bit different from the other datasets. The most notable characteristic is that there are not enough samples in some classes, a few of which have less than 20 labeled data. Therefore, we adopted the method of [30] to split the labeled data in a 1:1 ratio for training and testing.

The classification results are shown in Figures 6 and 7.

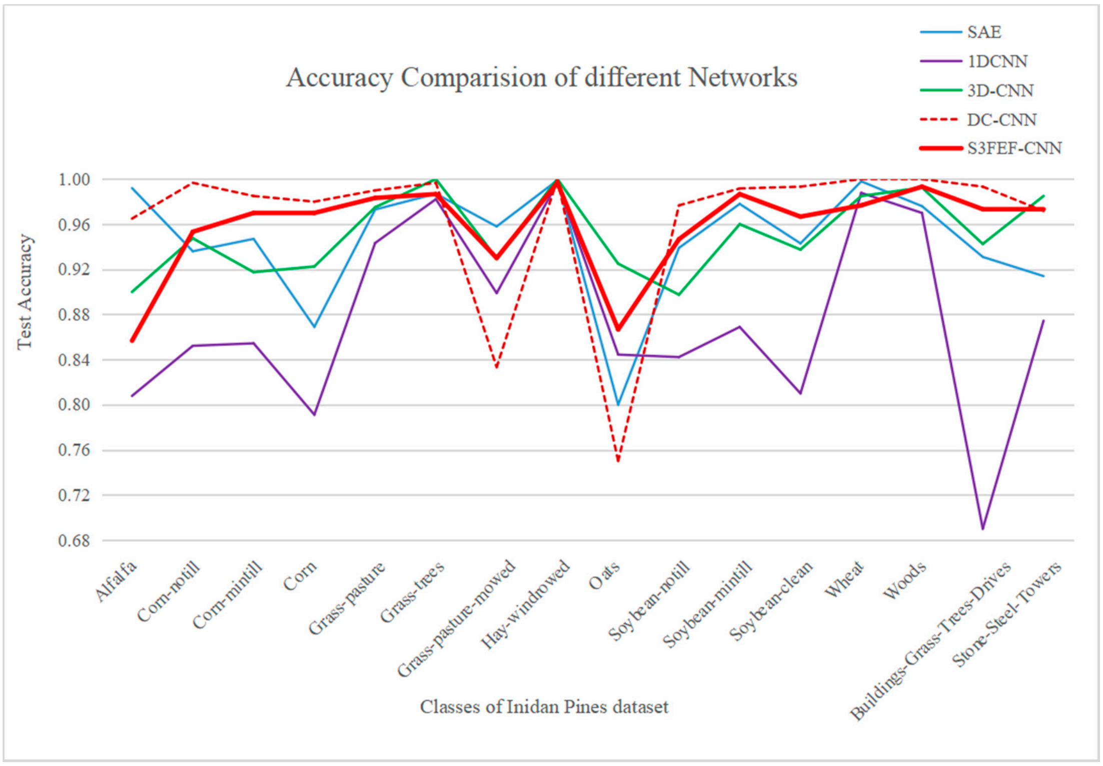


Figure 6. Overall accuracy on Indian Pines (IP) of different networks. It is clear that overall accuracy (OA) curve of S2FEF-CNN is much flatter than the others.

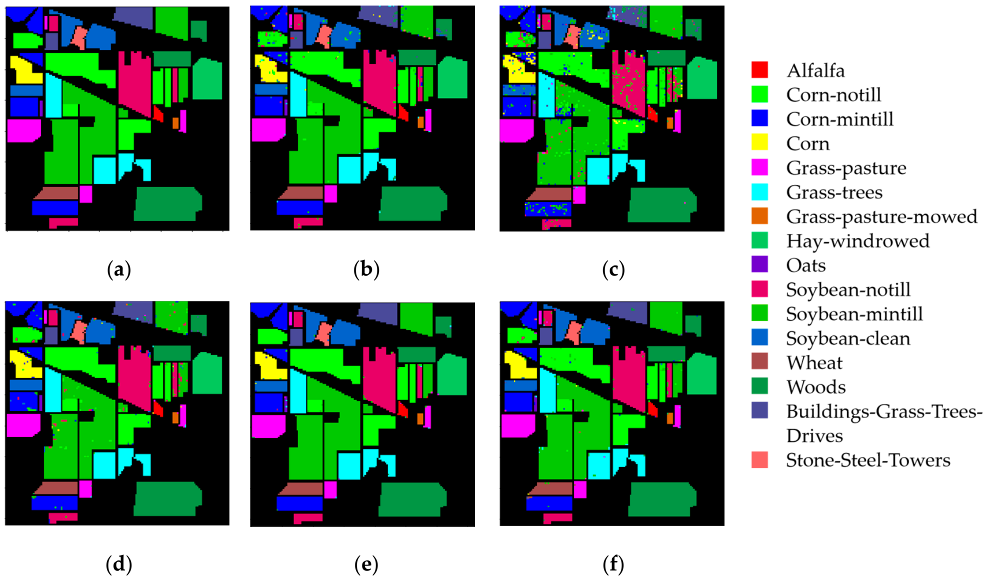


Figure 7. The classification map for Indian Pines dataset of different networks. (**a**) ground truth; (**b**) SAE (OA: 96.02%); (**c**) 1D-CNN (OA: 88.34%); (**d**) 3D-CNN (OA: 95.52%); (**e**) DC-CNN (OA: 99.11%); (**f**) S2FEF-CNN (OA: 97.43%).

In this paper, three commonly-used metrics were adopted to evaluate the classification performance, which are the overall accuracy (OA), the average accuracy (AA), and the Kappa coefficient. Figure 6 shows the OA curves with the visualized line diagram clearly.

From the results listed above, we can find that the S2FEF-CNN works well even with only a few thousand parameters. For all these methods, the OAs of 16 classes vary widely. For example, the class Oats has significantly lower OA because it has fewer labeled samples than the other categories.

4.5. Results of the Pavia University Dataset

In this dataset, there are enough labeled samples, so we set the split ratio to 2:8 for training and testing. The results are shown in Figures 8–10, where Figure 9 shows the training accuracy and loss curves.

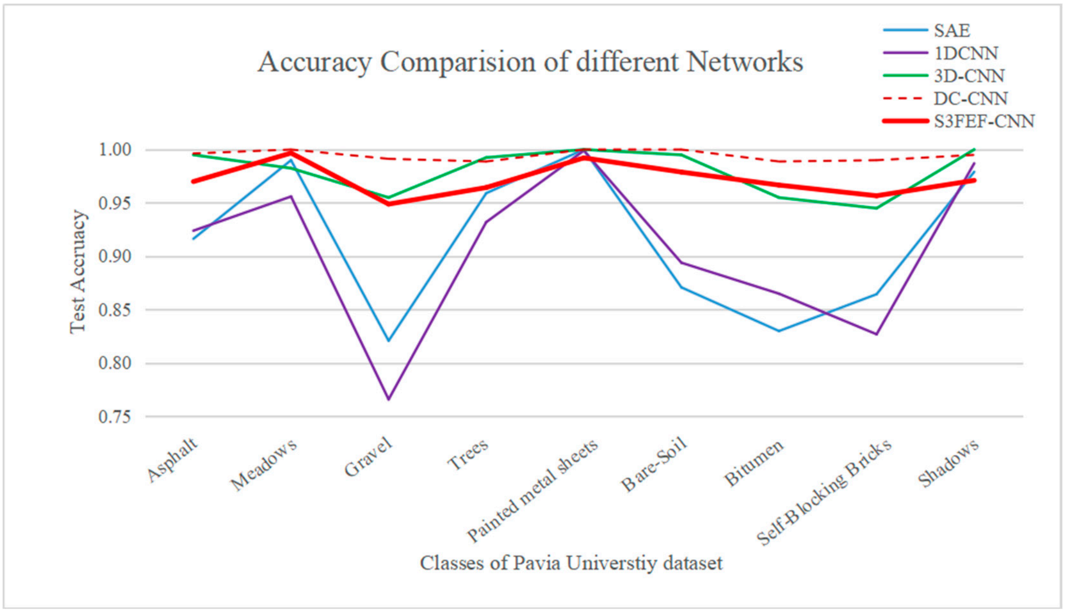
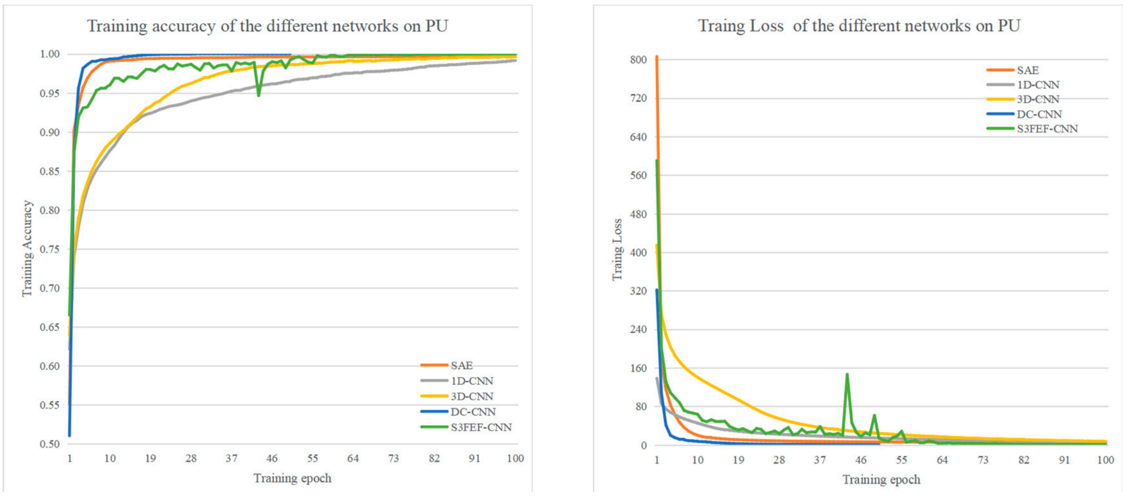


Figure 8. Overall accuracy on the Pavia University (PU) dataset of different networks. S2FEF-CNN works as well as it did on the IP dataset.



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

Figure 9. (**a**) Training accuracy curves on the PU dataset of different networks; (**b**) Loss curves on the PU dataset of different networks.

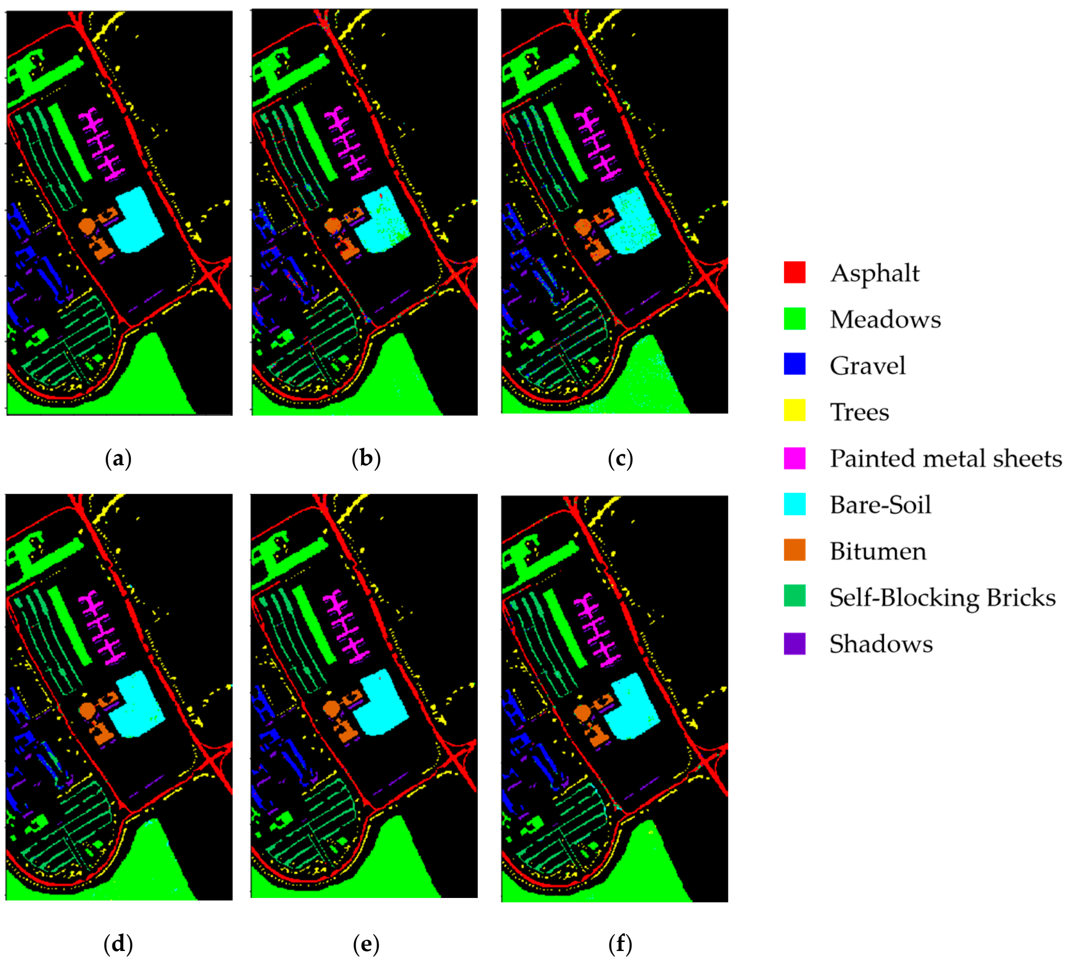


Figure 10. The classification map for the Pavia University dataset of different networks. (**a**) ground truth; (**b**) SAE (OA: 94.18%); (**c**) 1D-CNN (OA: 92.06%); (**d**) 3D-CNN (OA: 98.20%); (**e**) DC-CNN (OA: 99.66%); (**f**) S2FEF-CNN (OA: 98.04%).

As can be seen from Figure 8, S2FEF-CNN also performs well on the PU dataset, and it has a good classification ability for all nine classes. The accuracy of each class is more than 94%. There is little difference in the OA value between categories.

Figure 9 shows the dynamic changing of training on the PU dataset. All the methods converged after 100 epochs. DC-CNN and SAE are the fastest convergence methods. The curve of S2FEF-CNN oscillated a bity during the test.

4.6. Results of the Salinas Dataset

The portion of training set and test set of Salinas is the same as that of PU. Figures 11 and 12 show the results.

The performance of S2FEF-CNN on the SA dataset was similar to that on the IP and PU datasets, and the OA curve also looks stable. Several methods did not produce high results on the Grapes_untrained and Vinyard_untrained classes, and most of them were misclassified. From Figure 12 we can see that the two classes are very close geographically. In addition, the spectral lines of the two classes are also very similar. This may be the cause of the misclassification.

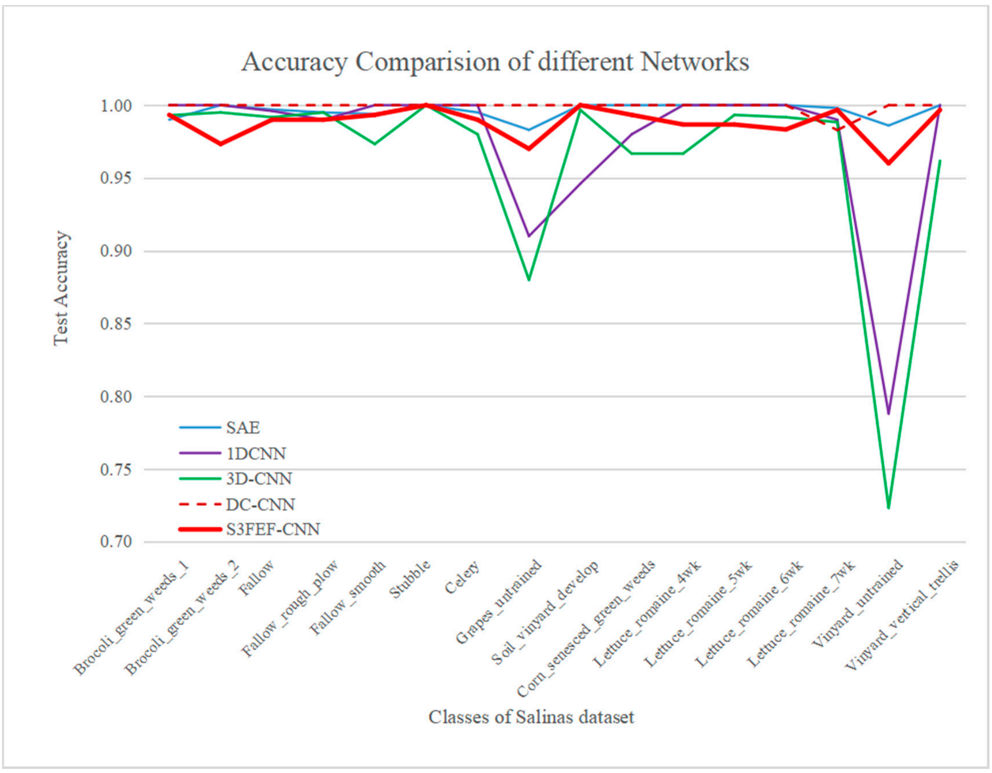


Figure 11. Overall accuracy on the Salinas dataset for different networks.

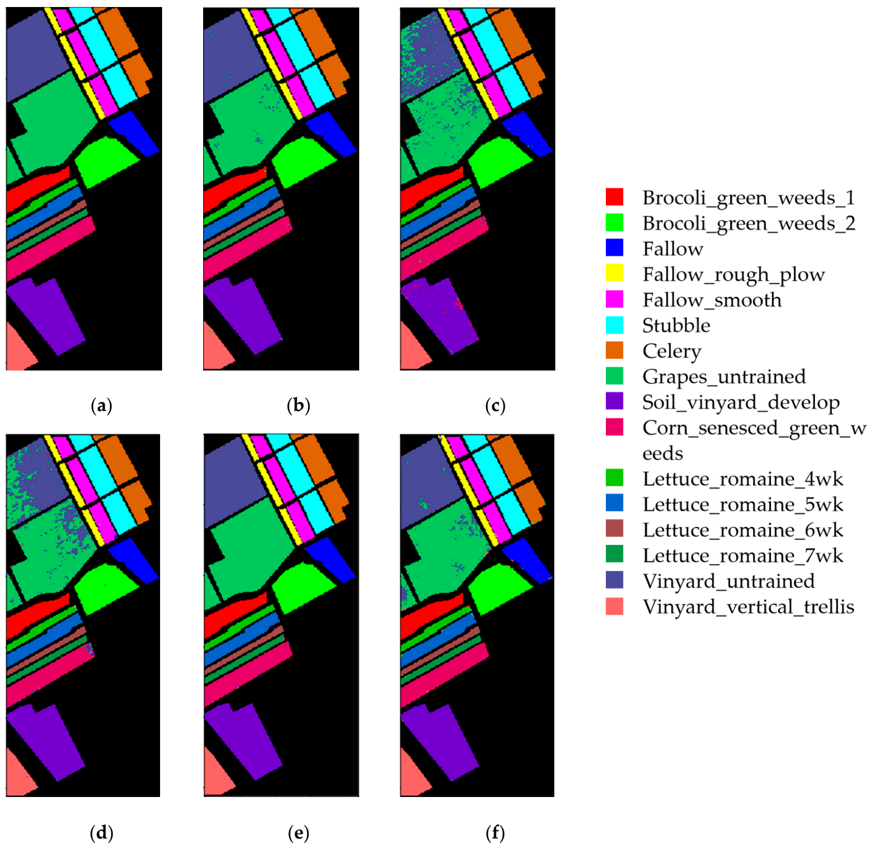


Figure 12. The classification map for the Salinas dataset of different networks. (**a**) ground truth; (**b**) SAE (OA: 99.29%); (**c**) 1D-CNN (OA: 94.46%); (**d**) 3D-CNN (OA: 92.87%); (**e**) DC-CNN (OA: 99.89%); (**f**) S2FEF-CNN (OA: 98.26%).

4.7. Parameter Influence

In this section, we discuss and show how the parameters influence the classification performance as shown in Figures 13–15. Some vital hyperparameters such as kernel number, kernel size, and network depth are discussed. We use the classification results of the Indian Pines dataset as an example.

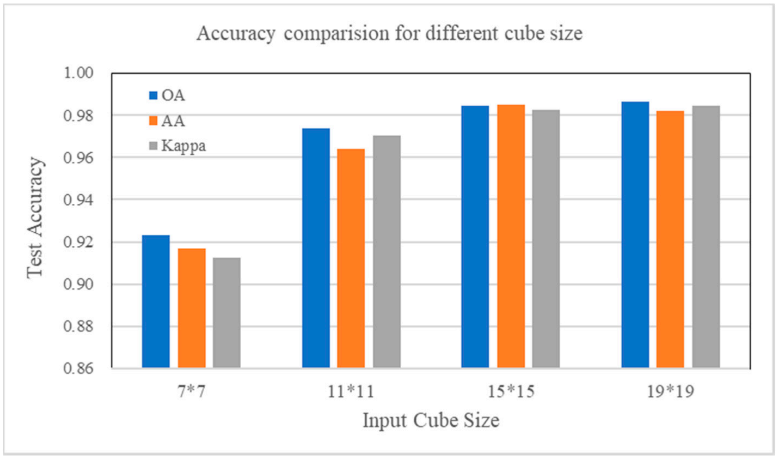


Figure 13. Overall accuracy, average accuracy, and Kappa of different cube spatial sizes.

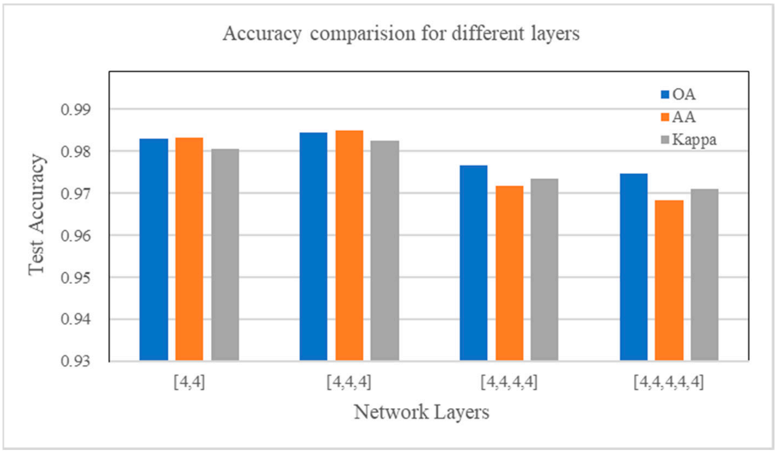


Figure 14. Overall accuracy, average accuracy, and kappa of variation of the S2FEF blocks (convolutional layers).

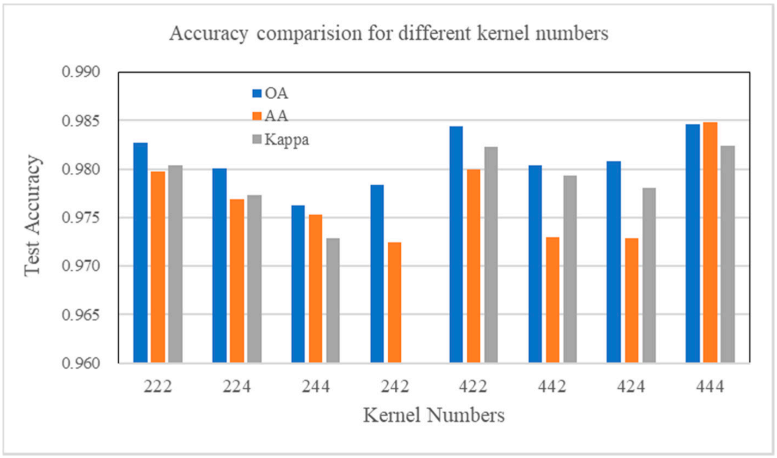


Figure 15. Overall accuracy, average accuracy, and kappa of variation of the number of the kernels.

Figure 13 shows the influence of cube size on OA, AA, and Kappa. As expected, when the spatial size of the cube increased from 7×7 to 19×19, the result increased significantly. Three-dimensional-CNN [30] uses 3D cubes as the input and sets the spatial size to 5×5. As mentioned before, we need a big spatial size of cube to extract sufficient features for the next classification. Of course, we made a tradeoff between performance and cost as well, and finally set the size to 19×19.

Figure 14 shows the results of different network depths. In general, the deeper the network the better, because a deeper network can extract more high-level features that are beneficial for classification. However, our results are not proportional to the depth of the network. In other words, deeper is not better in S2FEF architecture. Actually, networks with four or more blocks perform as well as that with three. Considering the balance between performance and cost, we eventually set the network layer to 3.

Another important parameter is the convolutional kernel numbers. Although there is no universal setting, 2^n is preferred. Figure 15 shows the accuracy comparison of different kernel numbers in each layer. We tried eight different combinations [2,2,2], [2,2,4], [2,4,4], [2,4,2], [4,2,2], [4,4,2], [4,2,4], and [4,4,4] (notation [k_1, k_2, k_3] means k_1 kernels in layer 1, k_2 kernels in layer 2, k_3 kernels in layer 3), and the combination of [4,4,4] worked the best. Unexpectedly, the suboptimal combination was [2,2,2].

5. Discussion

From the above results, we can draw the following conclusions.

Firstly, it is obvious that the deep network using spectral–spatial features can achieve better classification accuracy than those using only spectral features. The results strongly prove that spectral–spatial features benefit HSI classification.

Secondly, deep learning performs outstandingly in some remote sensing fields. However, the trend of making networks more complex and deeper brings a heavy load of parameters during training. More parameters may give the model better classification ability. It can be seen from the above results that DC-CNN shows the best accuracy. Sometimes we do not require high precision, but want to reduce the network parameters, so we can make some tradeoffs within an acceptable error range. Therefore, our method is a good attempt to simplify the network, such as with fewer kernels and/or fewer convolutional layers. PCA is an effective preprocessing method for dimension-reduction, while the fully connected layer is commonly used in CNN before classification, but these two common methods are replaceable. Our experimental results indicate that even simple and shallow networks can work well if we can come up with some effective strategies.

Finally, we have to talk about batch normalization, which is important for deep learning. Without batch normalization, the network can still work, but converges slowly. The explicit use of batch normalization forces the distribution of data to be more reasonable, which not only speeds up the convergence rate, but also smooths the accuracy curves.

6. Conclusions

In this paper, we explored an effective and novel spectral–spatial feature extraction and fusion block for HSI classification. Based on the block, we put forward a lightweight CNN-based network. Compared to state-of-the-art deep networks, the most impressive advantage of the proposed network is that it can obtain a considerable classification accuracy with very few parameters. This is a potentially feasible way to simplify the network while maintaining the accuracy of classification. Inevitably, we found various problems in training, e.g., convergence. Sometimes, bad initial variables led to very slow convergence or even nonconvergence. Computation of multiplication is also troublesome.

Our future research will focus on the following two points: 1) We will concentrate on an efficient fusion method of the spectral and spatial features to further reduce the computation; and 2) we will study the semi-supervised method for HSI classification. There is a lot of unlabeled data in HSI that may provide us with valuable features. The semi-supervised approach will look at how to make full use of these unmarked samples, which is worth studying.

Author Contributions: Methodology, L.C. and Z.W.; Software, L.C.; Writing—original draft, L.C.; Writing—review and editing, L.C., Z.W. and Y.X. All the authors contributed significantly to the research. All authors have read and agreed to the published version of the manuscript.

Funding: Our work was supported partly by the National Natural Science Foundation of China under Grant 61701238, Grant 61772274, Grant 11431015, and Grant 61671243, partly by the Jiangsu Provincial Natural Science Foundation of China under Grant BK20170858, Grant BK20180018, and Grant BK20191409, in part by the Fundamental Research Funds for the Central Universities under Grants 30919011234, 30917015104, 30919011103, and 30919011402, in part by the China Postdoctoral Science Foundation under Grant 2017M611814 and Grant 2018T110502, and in part by the Jiangsu Province Postdoctoral Science Foundation under Grant 1701148B.

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

References

1. Liang, L.; Di, L.; Zhang, L.; Deng, M.; Qin, Z.; Zhao, S.; Lin, H. Estimation of crop LAI using hyperspectral vegetation indices and a hybrid inversion method. *Remote Sens. Environ.* **2015**, *165*, 123–134. [CrossRef]
2. Tidke, S.; Kumar, A. New HyperSpectral Image Segmentation based on the Concept of Binary Partition Tree. *Int. J. Adv. Technol. Eng. Explor.* **2015**, *2*, 140–146.
3. Lu, X.; Li, X.; Mou, L. Semi-supervised multitask learning for scene recognition. *IEEE Trans. Cybern.* **2015**, *45*, 1967–1976. [PubMed]
4. Valero, S.; Salembier, P.; Chanussot, J. Object recognition in hyperspectral images using Binary Partition Tree representation. *Pattern Recognit. Lett.* **2015**, *56*, 4098–4101. [CrossRef]
5. Li, S.; Song, W.; Fang, L.; Chen, Y.; Ghamisi, P.; Benediktsson, J. Deep learning for hyperspectral image classification: An overview. *arXiv* **2019**, arXiv:1910.12861. [CrossRef]
6. Hecker, C.; Meijde, M.; Werff, H.; Meer, F. Assessing the influence of reference spectra on synthetic SAM classification results. *IEEE Trans. Geosci. Remote Sens.* **2008**, *46*, 4162–4172. [CrossRef]
7. Wang, K.; Gu, X.; Yu, T.; Meng, Q.; Zhao, L.; Feng, L. Classification of hyperspetral remote sensing images using frequency spectrum similarity. *Sci. China Tech. Sci.* **2013**, *56*, 980–988. [CrossRef]
8. Demir, B.; Erturk, S. Hyperspectral image classification using relevance vector machines. *IEEE Geosci. Remote Sens. Lett.* **2007**, *4*, 586–590. [CrossRef]
9. Camps-Valls, G.; Gomez-Chova, L.; Muñoz-Marí, J.; Vila-Francés, J.; Calpe-Maravilla, J. Composite kernels for hyper-spectral image classification. *IEEE Geosci. Remote Sens. Lett.* **2006**, *3*, 93–97. [CrossRef]
10. Gómez-Chova, L.; Camps-Valls, G.; Muoz-Mari, J.; Calpe, J. Semi-supervised image classification with Laplacian support vector machines. *IEEE Geosci. Remote Sens. Lett.* **2008**, *5*, 336–340. [CrossRef]
11. Licciardi, G.; Marpu, P.; Chanussot, J.; Benediktsson, J. Linear versus nonlinear PCA for the classification of hyperspectral data based on the extended morphological profiles. *IEEE Geosci. Remote Sens. Lett.* **2012**, *9*, 447–451. [CrossRef]
12. Li, W.; Chen, C.; Su, H.; Du, Q. Local binary patterns and extreme learning machine for hyperspectral imagery classification. *IEEE Trans. Geosci. Remote Sens.* **2015**, *53*, 3681–3693. [CrossRef]
13. Cao, X.; Xu, L.; Meng, D.; Zhao, Q.; Xu, Z. Integration of 3-dimensional discrete wavelet transform and markov random field for hyperspectral image classification. *Neurocomputing* **2017**, *226*, 90–100. [CrossRef]
14. Li, W.; Wu, G.; Zhang, F.; Du, Q. Hyperspectral image classification using deep pixel-pair features. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 844–853. [CrossRef]
15. Chen, Y.; Jiang, H.; Li, C.; Jia, X.; Ghamisi, P. Deep feature extraction and classification of hyperspectral images based on convolutional neural networks. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 6232–6251. [CrossRef]
16. Liang, J.; Zhou, J.; Qian, Y.; Wen, L.; Bai, X.; Gao, Y. On the sampling strategy for valuation of spectral-spatial methods in hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 862–879. [CrossRef]
17. Cheng, G.; Li, Z.; Han, J.; Yao, X.; Guo, L. Exploring hierarchical convolutional features for hyperspectral image classifification. *IEEE Trans. Geosci. Remote Sens.* **2018**, *56*, 6712–6722. [CrossRef]
18. Kang, X.; Li, C.; Li, S.; Lin, H. Classification of hyperspectral images by gabor filtering based deep network. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2018**, *11*, 1166–1178. [CrossRef]
19. Leng, J.; Li, T.; Bai, G.; Dong, Q.; Dong, H. Cube-CNN-SVM: A novel hyperspectral image classification method. In Proceedings of the IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI), San Jose, CA, USA, 6–8 November 2016; pp. 1027–1034.

20. Yu, D.; Deng, L.; Wang, S. Learning in the deep-structured conditional random fields. In Proceedings of the Conference and Workshop on Neural Information Processing Systems (NIPS), Vancouver, BC, Canada, 7–10 December 2009; pp. 1848–1852.
21. Mohamed, A.; Sainath, T.; Dahl, G.; Ramabhadran, B.; Hinton, G.E.; Picheny, M. Deep belief networks using discriminative features for phone recognition. In Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Prague, Czech, 22–27 May 2011; pp. 5060–5063.
22. Dieleman, S.; Schrauwen, B. End-to-end learning for music audio. In Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Florence, Italy, 4–9 May 2014; pp. 6964–6968.
23. Wang, T.; Wu, D.; Coates, A.; Ng, A. End-to-end text recognition with convolutional neural networks. In Proceedings of the 21st International Conference on Pattern Recognition (ICPR), Tsukuba, Japan, 11–15 November 2012; pp. 3304–3308.
24. Hu, W.; Huang, Y.; Li, W.; Zhang, F.; Li, H. Deep convolutional neural networks for hyperspectral image classification. *J. Sens.* **2015**, *2015*, 258619. [CrossRef]
25. Yue, J.; Zhao, W.; Mao, S.; Liu, H. Spectral-spatial classification of hyperspectral images using deep convolutional neural networks. *IEEE Geosci. Remote Sens. Lett.* **2015**, *6*, 468–477. [CrossRef]
26. Mei, S.; Ji, J.; Bi, Q.; Hou, J.; Du, Q.; Li, W. Integrating spectral and spatial information into deep convolutional neural networks for hyperspectral classification. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Beijing, China, 10–15 July 2016; pp. 5067–5070.
27. Haut, J.; Paoletti, M.; Plaza, J.; Li, J.; Plaza, A. Active learning with convolutional neural networks for hyperspectral image classification using a new bayesian approach. *IEEE Trans. Geosci. Remote Sens.* **2018**, *56*, 6440–6461. [CrossRef]
28. Liang, H.; Li, Q. Hyperspectral imagery classification using sparse representations of convolutional neural network features. *Remote Sens.* **2016**, *8*, 99. [CrossRef]
29. Makantasis, K.; Protopapadakis, E.; Doulamis, A.; Doulamis, N.; Loupos, C. Deep convolutional neural networks for efficient vision based tunnel inspection. In Proceedings of the IEEE International Conference on Intelligent Computer Communication and Processing (ICCP), Cluj-Napoca, Romania, 3–5 September 2015; pp. 335–342.
30. Li, Y.; Zhang, H.; Shen, Q. Spectral-spatial classification of hyperspectral imagery with 3d convolutional neural network. *Remote Sens.* **2017**, *9*, 67. [CrossRef]
31. 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. [CrossRef]
32. Chen, C.; Zhang, J.; Zheng, C.; Yan, Q.; Xun, L. Classification of hyperspectral data using a multi-channel convolutional neural network. In Proceedings of the 14th International Conference on Intelligent Computing (ICIC), Wuhan, China, 15–18 August 2018; pp. 81–92.
33. Paoletti, M.; Haut, J.; Plaza, J.; Plaza, A. A new deep convolutional neural network for fast hyperspectral image classification. *ISPRS J. Photogramm. Remote Sens.* **2018**, *145*, 120–147. [CrossRef]
34. He, M.; Li, B.; Chen, H. Multi-scale 3d Deep Convolutional Neural Network for Hyperspectral image classification. In Proceedings of the IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017; pp. 3904–3908.
35. Yang, J.; Zhao, Y.; Chan, J. Learning and transferring deep joint spectral-spatial features for hyperspectral classification. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 4729–4742. [CrossRef]
36. Zhang, H.; Li, Y.; Zhang, Y.; Shen, Q. Spectral-spatial classification of hyperspectral imagery using a dual-channel convolutional neural network. *IEEE Geosci. Remote Sens. Lett.* **2017**, *8*, 438–447. [CrossRef]
37. Yang, J.; Zhao, Y.; Chan, J.; Yi, C. Hyperspectral image classification using two-channel deep convolutional neural network. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Beijing, China, 10–15 July 2016; pp. 5079–5082.
38. Roy, S.; Krishna, G.; Dubey, S.; Chaudhuri, B. HybridSN: Exploring 3D-2D CNN feature hierarchy for hyperspectral image classification. *arXiv* **2019**, arXiv:1902.06701v2.
39. 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]
40. Chen, Y.; Zhao, X.; Jia, X. Spectral-spatial classification of hyperspectral data based on deep belief network. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2015**, *8*, 2381–2392. [CrossRef]

41. Wu, Q.; Diao, W.; Dou, F.; Sun, X.; Zheng, X.; Fu, K.; Zhao, F. Shape-based object extraction in high-resolution remote-sensing images using deep Boltzmann machine. *Int. J. Remote Sens.* **2016**, *37*, 6012–6022. [CrossRef]
42. Deng, F.; Pu, S.; Chen, X.; Shi, Y.; Yuan, T.; Pu, S. Hyperspectral image classification with capsule network using limited training samples. *Sensors* **2018**, *18*, 3153. [CrossRef] [PubMed]
43. Kang, X.; Zhuo, B.; Duan, P. Dual-path network-based hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **2019**, *16*, 447–451. [CrossRef]
44. Hamida, A.; Benoit, A.; Lambert, P.; Amar, C. 3-D deep learning approach for remote sensing image classification. *IEEE Trans. Geosci. Remote Sens.* **2018**, *56*, 4420–4434. [CrossRef]