IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 11, 2019, accepted June 20, 2019, date of publication June 26, 2019, date of current version July 15, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2925283

SSDC-DenseNet: A Cost-Effective End-to-End Spectral-Spatial Dual-Channel Dense Network for Hyperspectral Image Classification

YUTONG BAI, QIFAN ZHANG, ZEXIN LU, AND YI ZHANG, (Senior Member, IEEE)
College of Computer Science, Sichuan University, Chengdu 610065, China

Corresponding author: Yi Zhang (yzhang@scu.edu.cn)

This work was supported in part by the National Key R&D Program of China under Grant 2018YFC0830300, in part by the National Natural Science Foundation of China under Grant 61671312 and in part by the Sichuan Science and Technology Program under Grant 2018HH0070.

ABSTRACT In recent years, various deep learning-based methods have been applied in hyperspectral image (HSI) classification. Among them, spectral-spatial approaches have demonstrated their power to yield high accuracies. However, these methods tend to be computationally expensive. Specifically, two classic ways to develop spectral-spatial approaches both suffer from significant limitations in cost reduction: multi-channel networks need a large parameter scale, and 3-D filters are inherent of computational complexity. To establish a cost-effective architecture for both training cost and parameter scale, while maintaining the high accuracy of spectral-spatial techniques, an end-to-end spectral-spatial dual-channel dense network (SSDC-DenseNet) is proposed. To explore high-level features, the densely connected structure is introduced to enable deeper network. Furthermore, a 2-D deep dual channel network is applied to replace the expensive 3-D filters to reduce the model scale. The experiments were conducted on three popular datasets: the Indian Pines dataset, University of Pavia dataset, and Salinas dataset. The results demonstrate the competitive performance of the proposed SSDC-DenseNet with respect to classification performance and computational cost compared with other state-of-the-art DL-based methods while obtaining a remarkable reduction of computational cost.

INDEX TERMS Deep learning, densely connected convolutional neural network, feature extraction, hyperspectral image classification, multi-scale filter bank.

I. INTRODUCTION

Hyperspectral images (HSIs) are characterized by hundreds of narrow bands, spanning from the visible to infrared spectrum. The detailed spectral information increases the likelihood of discriminating ground objects of interest.

As hyperspectral remote sensors become increasingly powerful, the dimension of remote sensing data continually increases. Simultaneously, the number of publicly available HSI datasets continues to grow. The above improvements make the analysis of HSI data a critical technique in practical applications, such as precision agriculture, environmental monitoring [1], [2] and ocean research [3], [4]

However, because of the growing complexity of spectral information and the scarcity of labeled training samples, HSI classification, which refers to pixel-wise labeling of the spectrum, becomes more meaningful and challenging.

Favuel *et al*. [5] concluded that there are two major challenges in HSI classification: spectral dimensionality and the need for spectral-spatial classifiers. Extensive efforts have been made in this area, including feature selection [6], [7] and extraction techniques, such as Bayesian models [8], principal component analysis (PCA) [9], independent component analysis [10] and manifold learning [11], in addition to classifiers, such as support vector machines [12] and decision trees [13]. However, inappropriate dimensionality reduction in the spectral domain may fail to fully exploit the discriminative features, and conventional classifiers, which typically consider HSI as a list of wavelength measurements without spatial organization, cannot make the classification accuracy acceptable. Therefore, more effective methods to characterize spectral-spatial signatures should be developed.

The associate editor coordinating the review of this manuscript and approving it for publication was Dong Wang.

In recent decades, deep learning (DL) methods have become popular in the field of machine learning to solve the image classification problem [14], [15], and have outperformed many traditional classification models. DL provides an end-to-end approach that can extract features and classify simultaneously without a handcrafted feature extractor. Additionally, DL makes it possible to learn parameters automatically and hierarchically to extract discriminative and robust features. This breakthrough regarding DL has encouraged researchers to adopt it into HSI classification. Stacked autoencoders (SAEs) [16], deep belief networks [17], [18] and convolutional neural networks (CNNs) [19]–[21] have been used in this field. However, some limitations have been demonstrated for parts of these methods. For example, the stacked autoencoders with logistic regression (SAE-LR) [22] based method requires PCA, which is quite time-consuming, to extract features before classification. Because CNNs use fewer parameters than fully connected (FC) networks to exploit the nonlinear features of HSI, they can efficiently alleviate the overfitting problem caused by limited training samples.

As mentioned by Favuel *et al.* [5], the first challenge is the large number of spectral bands. Recently, multiple CNN-based methods have been focusing on deepening the network to extract high-level features, which aggravates the need for alleviating the overfitting problem. For instance, inspired by the deep residual network (ResNet) [23], which proposed skip connections to facilitate the propagation of gradients. Zhong *et al.* [24] introduced ResNet for HSI classification. Compared with traditional CNN models with the same number of layers, ResNet achieved better performance for HSI classification. However, it should be noted that Zhong *et al.*'s model still suffers from performance degradation when it goes deeper. Moreover, transferring features strongly depends on the network design, particularly where and how to insert the skip connections.

In the meantime, attention was paid on the second challenge: the design of effective spectral-spatial classifiers. For the CNN-based methods, in the early stage, efforts were mainly made in the spectral domain. For examples, in 2015, Hu *et al.* [19] applied a CNN to extract spectral features, and the results outperformed those of an SVM. Soon, attention was drawn to the spatial domain. Slavkovikj *et al.* [20] constructed a CNN architecture that took neighboring pixels into consideration. Their work demonstrated that spatial features are great complements to spectral features. In fact, by training the network with high-order data (three-order tensor), a CNN has the inherent potential to simultaneously learn spatial and spectral features. To further use detailed spectral-spatial information, numerous types of CNN-based classification models have been proposed. For example, Yang *et al.* [25] proposed a two-channel deep CNN (two-CNN). Similarly, Chen *et al.* [26] presented a regularized three-dimensional (3D) CNN-based feature extraction model. In 2017, contextual CNN [27] with a multi-scale filter bank was proposed to use spectral-spatial information. Moreover, attempts have been made in designing effective 3D spectral-spatial models. Wang *et al.* [28] proposed a fast dense spectral-spatial convolution (FDSSC) framework with a dense structure [29]. In [28], the concatenations of the feature-maps fully take advantage of the extracted features from the previous layers and efficiently avoid gradient vanishing. In spite of high accuracy, FDSSC is of a large scale and requires high computational resources. Although these spectral-spatial models have demonstrated great powers for HSI classification, especially for deep networks which extract high-level features from a great number of bands, the model scale is huge and will significantly extend the training time. To circumvent this obstacle while maintaining the high accuracy, in this paper, we aim at designing a cost-effective deep model for HSI classification.

There are two popular ways to exploit spectral-spatial features: multi-channel networks and 3D filters. However, both methods could suffer from inevitable limitations of computational cost. Many of the multi-channel networks [27], [30] work on widening and deepening the network to exploit high-level features, which leads to a large parameter scale that may cause the overfitting problem and preventing the network from further accuracy improvement. For 3D networks, the inherent computational complexity of 3D filters will heavily enlarge the number of parameters and lead to a huge model scale with considerable training cost.

To deal with problems mentioned above, in this paper, we propose a spectral-spatial dual-channel dense network (SSDC-DenseNet). Our attempts are mainly made in two aspects: (1) dense connections are adopted in both dual-channel part and fusion part, which facilitates the information flow to effectively learn high-level features, as well as alleviate the overfitting problem; and (2) 2D filters are utilized to replace 3D ones for model scale reduction.

The remainder of this paper is organized as follows: In Section II, we provide a description of the proposed framework. In Section III, we present the experiments and analysis. We first conduct and discuss extensive experiments to compare it with several state-of-the-art methods on three real-world HSI datasets. We further analyze the effectiveness on the cost of 2D dense modules compared to 3D ones. Finally, we draw conclusions in Section IV.

II. PROPOSED FRAMEWORK

In this section, we first provide an overview of the proposed SSDC-DenseNet architecture, and then elaborate on the data preprocessing strategy and the structure of each block. The input data size of the network is $p \times p \times N$, where p is the patch size of the input image and N is its number of channels. The output is a $1 \times C$ vector, where C is the number of classes. With the help of the softmax classifier, the network performs pixel-wise classification in an end-to-end manner.

A. OVERVIEW OF THE ARCHITECTURE

The network structure of SSDC-DenseNet is illustrated in Fig. 1. It is composed of three cascaded blocks: a spectral feature extraction block, joint spectral-spatial feature learning

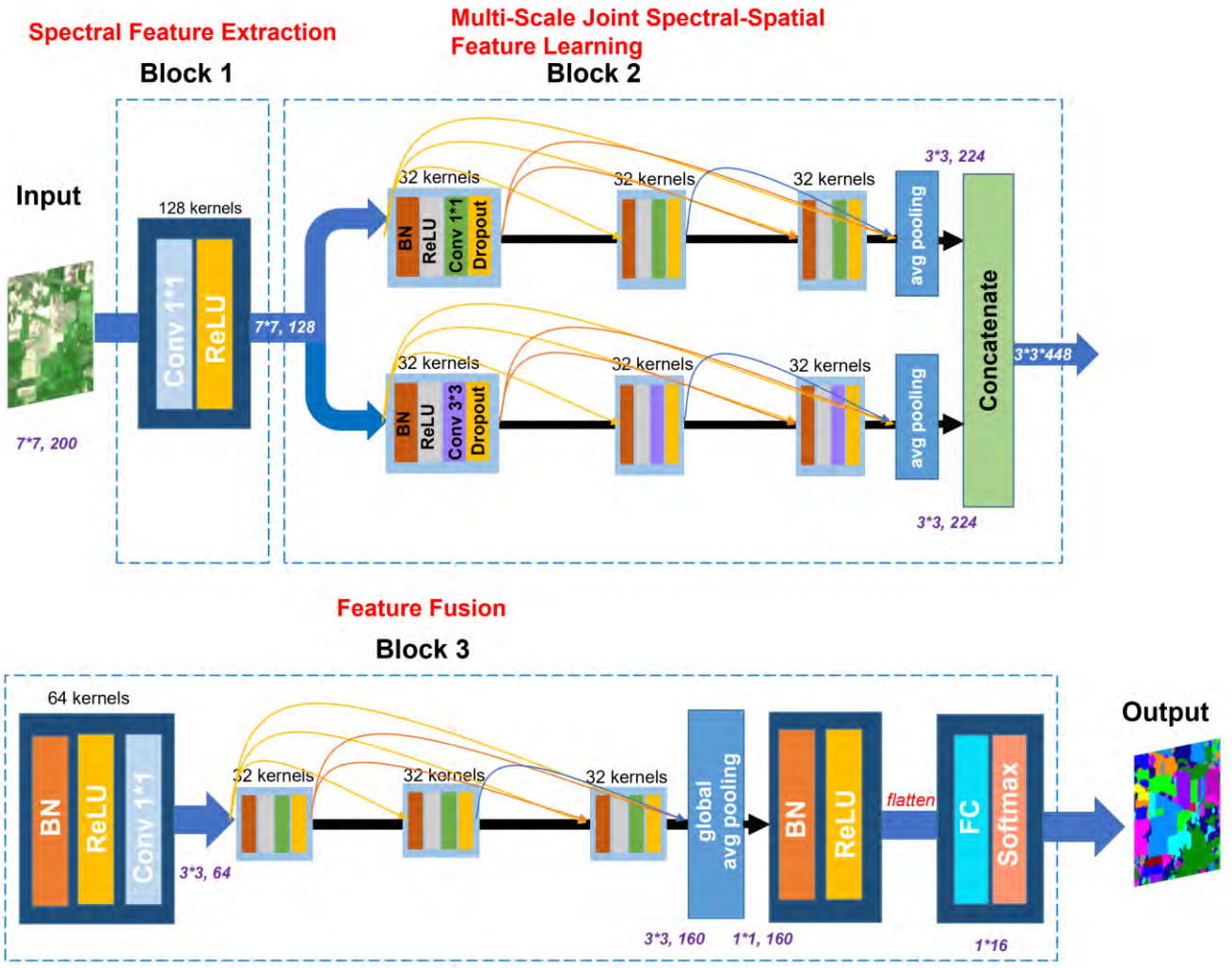

FIGURE 1. Overall structure of the proposed SSDC-DenseNet.

block using a multi-scale filter bank, and feature fusion block. Here we take the input as IN dataset with 7×7 patch to illustrate the size of feature maps. Note that the kernel numbers per layer in Block 2 vary for different datasets, which will be further illustrated in Section III.C and Section III.E.

B. DATA PREPROCESSING

To take advantage of both spectral and spatial information, we feed the network with a window containing a small neighborhood of pixels. The desired label for the network is the class of the centered pixel. To do this, we first split the input image into n patches of size, where p is the width and height of the input vector and N is the total number of bands in the original hyperspectral image. If the pixels on the border cannot be attributed to any neighborhood, the $p/2$ pixels on the border are mirrored outward to create corresponding patches.

C. BLOCK 1: SPECTRAL FEATURE EXTRACTION

Due to the importance of spectral information for HSIs, spectral and spectral-spatial information are extracted jointly in two blocks in the proposed network.

In Block 1, a convolution layer is used, followed by a rectified linear unit (ReLU) [30]. This configuration serves to limit the number of feature maps sent to Block 2 in a reasonable range through the setting of kernel number, as well as retain the spatial information of HSIs while exploiting the spectral information.

Let N^k be the number of filters in layer k and $\boldsymbol{X}_j^k$ denote the jth input feature tensor; $*$ is the convolution operator; and $\boldsymbol{H}_i^{k+1}$ and b_i^{k+1} are the ith filter and bias in layer $k+1$, respectively. Output $\boldsymbol{X}_i^{k+1}$ of the convolution operation is

$$\boldsymbol{X}_i^{k+1} = \sum_{j=1}^{N^k} \boldsymbol{X}_j^k * \boldsymbol{H}_i^{k+1} + b_i^{k+1} \tag{1}$$

The ReLU function is defined as

$$\tilde{\boldsymbol{Z}} = \max\{0, \boldsymbol{Z}\}, \tag{2}$$

where $\boldsymbol{Z}$ is the input tensor.

The filter size in this layer is set to $1 \times 1 \times N$, and the 1×1 spatial size is set to exploit spectral information.

After Block 1, the number of the output channels (spectral bands) will be changed according to the number of kernels, while the value of each channels is generated from all the input channels as follows

$$\boldsymbol{X}_i^{k+1} = \sum_{j=1}^{N^k} \boldsymbol{X}_j^k w_i^{k+1} + b_i^{k+1} \tag{3}$$

Equation (3) explains the spectral feature extraction operation of 1×1 spatial convolution. The spectral bands equal to the input channels. w_i^{k+1} is the coefficient of the corresponding 1×1 filter. We can see that the value of each output channel is generated from all the input channels while the number of output channels is determined by the number of kernels. Meanwhile, due to the usage of 1×1 filters, the whole procedure of convolution can achieve the same effect as fully connected layer [27] and the computation is only along the spectral dimension (it also can be seen as convolving with a $1 \times 1 \times N$ filter and N denotes the number of bands), so Block 1 can extract spectral features.

Then, the network is fed with discriminative spectral features and well-preserved spatial features, which makes the subsequent spectral-spatial feature learning process easier.

After the feature extraction, features are sent to Block 2, which is a multi-scale dual-channel convolutional sub-network.

D. BLOCK 2: MULTI-SCALE JOINT SPECTRAL-SPATIAL FEATURE LEARNING

As mentioned in Section I, to avoid the expensive computational cost of 3D filters, 2D filters are potential substitution. However, the performances of 2D networks are often weaker for jointly spectral-spatial features excavation than 3D ones. Hence, to balance the computational cost and accuracy, we construct a 2D dual-channel component to learn the spectral-spatial features simultaneously. In Block 2, the feature maps are first sent to a multi-scale dual-channel convolutional sub-network, and then concatenated for Block 3.

Block 2 is inspired by GoogLeNet proposed in [31]. In this sub-network, the information is parallelly sent to convolutional filters with different scales. Although GoogLeNet is effective in multiple computer vision tasks, some adaptations have to be made to achieve desirable performance for HSI characteristics. Two modifications are committed to the original GoogLeNet.

The first modification is the filter size: 1×1 convolutional filters are adopted to focus on the spectral features while maintaining the abundant spatial information.

Additionally, 3×3 filters are adopted to exploit spatial features. From (1), we can see that the 3×3 convolution operation can addresses the spatial correlation of neighboring pixels.

The second modification is the depth of the filter bank. Deep networks are crucial to extract high-level features of HSI. However, considering the lack of labeled training data, dense blocks are used instead of simply stacked convolutional layers. The general structure of a dense block with three layers is shown in Fig. 2. All the feature maps produced by the preceding $(l-1)$ layers are combined using concatenation and fed into the lth layer. This procedure can be formulated as [29]

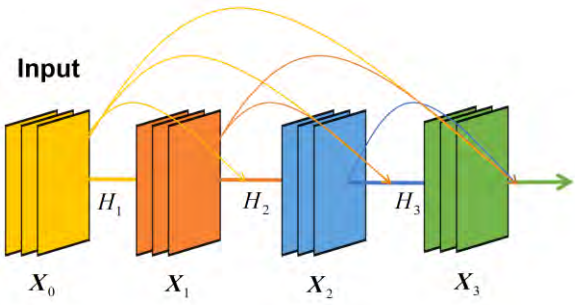

FIGURE 2. General structure of a dense block with three layers.

$$\boldsymbol{X_l} = H_l\left(\left[\boldsymbol{X}_0, \boldsymbol{X}_1, \ldots, \boldsymbol{X_{l-1}}\right]\right), \quad l \in N^+ \tag{4}$$

where $\left[\boldsymbol{X}_0, \boldsymbol{X}_1, \ldots, \boldsymbol{X_{l-1}}\right]$ denotes the concatenation of the output feature maps after layers $0, 1, \ldots, l-1$. In the proposed SSDC-DenseNet, $H_l(\cdot)$ is the composite function defined as four consecutive operations: batch normalization (BN) [32], ReLU, convolution and dropout [14].

BN is illustrated as [32]

$$\hat{\boldsymbol{X}}^{\boldsymbol{k}} = \frac{\boldsymbol{X^k} - \mathrm{E}\left(\boldsymbol{X^k}\right)}{\mathrm{VAR}\left(\boldsymbol{X^k}\right)} \tag{5}$$

where $\boldsymbol{X^k}$ denotes the kth layer's batch feature maps and $\mathrm{E}\left(\boldsymbol{X^k}\right)$ denotes the expectation of $\boldsymbol{X^k}$, Similarly, $\mathrm{VAR}\left(X^k\right)$ is the variance of $\boldsymbol{X^k}$ and $\hat{\boldsymbol{X}}^{\boldsymbol{k}}$ is the normalization result of the input tensor. This strategy enables deep neural networks to converge more smoothly.

Subsequently, the BN results are convolved with 1×1 or 3×3 after applying ReLU as the activation function.

To further address the overfitting problem, we add a dropout operation after ReLU. This operation sets the output maps generated by some neurons to zero with a specific probability. In this paper, the dropout rate is set to 0.5, which is commonly used in DL models. Subsequently, an average pooling layer is used with the kernel size 3 and the stride 2.

In the end, the generated feature maps are concatenated and fed to Block 3, which finally fuses the extracted spectral-spatial features.

E. BLOCK 3: FEATURE FUSION

In this block, the results of spectral and spatial learning are concatenated as input followed by a convolution layer and dense block, which is the same as the process for Block 2.

At the end of the dense block, global average pooling layers are inserted. The concept of global average pooling was first proposed in [33]. It was originally designed to replace the traditional FC layer in CNN. The average of the feature maps is calculated in this layer.

The global average pooling layer contains a much smaller number of parameters than FC layers and can retain

TABLE 1. Land cover classes with their respective sample numbers for the IN dataset.

#	Class	No. of samples
1	Alfalfa	46
2	Corn-notill	1428
3	Corn-mintill	830
4	Corn	237
5	Grass-pasture	483
6	Grass-trees	730
7	Grass-pasture-mowed	28
8	Hay-windrowed	478
9	Oats	20
10	Soybean-notill	972
11	Soybean-mintill	2455
12	Soybean-clean	593
13	Wheat	205
14	Woods	1265
15	Buildings-Grass-Trees-Drives	386
16	Stone-Steel-Towers	93
	Total	10,249

TABLE 2. Land cover classes with their respective sample numbers for the UP dataset.

#	Class	No. of samples
1	Asphalt	6631
2	Meadows	18,649
3	Gravel	2099
4	Trees	3064
5	Painted metal sheets	1345
6	Bare soil	5029
7	Bitumen	1330
8	Self-blocking bricks	3682
9	Shadows	947
	Total	42,766

TABLE 3. Land cover classes with their respective sample numbers for the Salinas dataset.

#	Class	No. of samples
1	Brocoli-green-weeds-1	2009
2	Brocoli-green-weeds-2	3726
3	Fallow	1976
4	Fallow-rough-plow	1394
5	Fallow-smooth	2678
6	Stubble	3959
7	Celery	3579
8	Grapes-untrained	11,271
9	Soil-vineyard-develop	6203
10	Corn-senesced-green-weeds	3278
11	Lettuce-romaine-4wk	1068
12	Lettuce-romaine-5wk	1927
13	Lettuce-romaine-6wk	916
14	Lettuce-romaine-7wk	1070
15	Vineyard-untrained	7268
16	Vineyard-vertical-trellis	1807
	Total	54,129

remarkable localization ability for a network. It is efficient to consider two main problems in HSI classification: the overfitting phenomenon caused by the large model scale with limited training data, and the effective extraction for both spectral and spatial features.

After the FC layer, a softmax layer is used to obtain the final classification result.

TABLE 4. Classification accuracy (%) and K measure comparison of the proposed SSDC-DenseNet and other DL methods: (a) IN, (b) UP and (c) Salinas.

Method	OA(%)	AA(%)	K × 100
SAE-LR [22]	94.12 ± 0.83	92.30 ± 1.59	93.29 ± 0.94
Conv-4 [24]	93.72 ± 0.74	84.66 ± 1.61	92.82 ± 0.85
ResNet-4 [24]	98.63 ± 2.45	93.00 ± 1.73	98.44 ± 0.28
contextual CNN [27]	95.09 ± 0.86	90.13 ± 1.75	94.40 ± 0.98
FDSSC [28]	99.41 ± 0.08	99.24 ± 0.24	99.33 ± 0.09
SSDC-DenseNet	**99.47 ± 0.17**	**99.30 ± 0.29**	**99.39 ± 0.20**

(a)

Method	OA(%)	AA(%)	K × 100
SAE-LR	98.97 ± 0.11	98.67 ± 0.12	98.66 ± 0.10
Conv-4	98.73 ± 0.13	97.69 ± 0.21	98.32 ± 0.17
ResNet-4	99.51 ± 0.13	99.18 ± 0.21	99.36 ± 0.17
contextual CNN	98.45 ± 0.17	97.50 ± 0.33	97.95 ± 0.23
FDSSC	99.75 ± 0.08	99.54 ± 0.16	99.62 ± 0.11
SSDC-DenseNet	**99.84 ± 0.06**	**99.74 ± 0.11**	**99.79 ± 0.08**

(b)

Method	OA(%)	AA(%)	K × 100
SAE-LR	98.40 ± 0.54	99.25 ± 0.28	98.21 ± 0.60
Conv-4	97.43 ± 0.17	98.61 ± 0.14	97.16 ± 0.22
ResNet-4	98.68 ± 0.18	99.31 ± 0.11	98.53 ± 0.20
contextual CNN	97.14 ± 0.33	98.47 ± 0.15	96.81 ± 0.37
FDSSC	99.30 ± 0.14	99.63 ± 0.07	99.22 ± 0.16
SSDC-DenseNet	**99.73 ± 0.07**	**99.82 ± 0.06**	**99.70 ± 0.08**

(c)

The main differences between the proposed SSDC-DenseNet and FDSSC are as follows:

1. FDSSC uses two different dense blocks (a spectral block followed by a spatial block) to extract features and spectral features sequentially, which makes the network require a specific depth for two-stage learning. This part may fail to extract features effectively, particularly for the second block, where the original information may have been altered by previous operations. To circumvent this obstacle, a dual-channel architecture is adopted to learn spectral and spatial features jointly.
2. FDSSC directly sends the feature maps generated by its second block to an average pooling layer and performs the softmax regression. Before the regression layer, a dense block is inserted to further extract the features from the results of joint learning in the proposed SSDC-DenseNet.
3. To reduce the network complexity and amount of computational resources of FDSSC, 3D filters are replaced by two-dimensional filters in SSDC-DenseNet. Appropriate operations are applied for the kernel sizes of filters to implement our strategy.

III. EXPERIMENTS AND ANALYSIS

In this section, we provide a detailed description of the datasets and configurations of the experiments. To validate

TABLE 5. Classification accuracy (%) of each class: (a) IN, (b) UP and (c) Salinas.

Class	SAE-LR	Conv-4	ResNet-4	contextual CNN	FDSSC	SSDC-DenseNet
1	100.0	47.06	82.35	73.53	100.0	97.06
2	93.06	91.78	98.04	91.41	98.32	99.44
3	89.82	95.98	99.68	94.21	99.20	99.04
4	95.56	86.44	100.0	87.01	99.44	98.31
5	97.98	97.51	98.62	97.51	98.07	99.45
6	98.84	99.27	99.45	100.0	99.45	99.45
7	100.0	71.43	90.48	80.95	100.0	100.0
8	100.0	98.60	100.0	100.0	100.0	100.0
9	66.67	73.33	20.00	73.33	100.0	100.0
10	91.49	96.02	98.90	95.20	99.86	99.31
11	94.23	95.87	98.15	96.69	99.95	99.67
12	92.92	82.21	97.97	84.23	98.20	99.10
13	97.92	99.35	100.0	99.35	99.35	100.0
14	97.62	98.42	99.79	97.79	99.47	100.0
15	75.38	95.16	99.31	87.89	97.58	100.0
16	92.86	75.36	100.0	100.0	100.0	98.55
OA(%)	94.15	94.61	98.62	94.79	99.35	99.53
AA(%)	92.77	87.74	92.67	91.19	99.39	99.41
K × 100	93.33	93.85	98.43	94.06	99.26	99.47

(a)

Class	SAE-LR	Conv-4	ResNet-4	contextual CNN	FDSSC	SSDC-DenseNet
1	98.79	98.40	99.79	98.57	100.0	100.0
2	99.57	99.85	100.0	99.79	100.0	100.0
3	97.89	95.47	95.77	93.27	96.84	99.11
4	99.02	98.57	98.41	98.57	99.51	98.94
5	100.0	99.81	99.91	100.0	100.0	100.0
6	98.92	99.75	100.0	99.68	100.0	99.95
7	98.16	92.39	97.84	94.46	100.0	99.72
8	97.96	96.67	98.07	95.04	98.95	99.73
9	100.0	99.74	99.74	100.0	100.0	100.0
OA(%)	99.08	98.80	99.40	98.62	99.72	99.84
AA(%)	98.92	97.85	98.84	97.71	99.48	99.72
K × 100	98.79	98.41	99.21	98.17	99.63	99.79

(b)

Class	SAE-LR	Conv-4	ResNet-4	contextual CNN	FDSSC	SSDC-DenseNet
1	99.73	100.0	100.0	99.94	100.0	100.0
2	100.0	99.97	99.90	99.87	100.0	99.60
3	99.74	99.56	100.0	98.86	100.0	100.0
4	100.0	99.91	99.46	99.01	99.19	99.19
5	99.82	99.72	100.0	99.63	100.0	100.0
6	100.0	100.0	100.0	100.0	100.0	100.0
7	100.0	99.72	100.0	99.86	99.93	100.0
8	98.57	94.84	97.15	92.56	98.85	99.67
9	100.0	99.94	99.98	99.82	99.98	100.0
10	99.84	97.29	98.25s	96.99	99.85	99.77
11	100.0	99.30	99.88	96.49	99.65	100.0
12	100.0	100.0	100.0	100.0	100.0	100.0
13	100.0	100.0	100.0	100.0	100.0	100.0
14	100.0	97.90	99.88	99.65	99.77	99.77
15	92.90	92.43	95.99	91.97	96.90	98.93
16	100.0	98.96	99.24	98.48	99.17	99.79
OA(%)	98.72	97.60	98.71	96.94	99.27	99.73
AA(%)	99.41	98.72	99.36	98.32	99.58	99.81
K × 100	98.58	97.32	98.56	96.59	99.19	99.70

(c)

TABLE 6. Time consumption of different DL methods for the three datasets: (a) IN, (b) UP and (c) Salinas.

Method	Training Time	Testing time
SAE-LR	56.134 m ± 0.461m	0.01 s ± 0.003 s
Conv-4	92.96 s ± 20.15 s	0.87 s ± 0.01 s
ResNet-4	94.05 s ± 19.80 s	0.85 s ± 0.01 s
contextual CNN	320.46 s ± 52.58 s	1.58 s ± 0.01s
FDSSC	1655.38 s ± 333.67 s	8.16 s ± 0.01 s
SSDC-DenseNet	484.11 s ± 65.16 s	1.55 s ± 0.21 s

(a)

Method	Training Time	Testing time
SAE-LR	400.62 m ± 2.24 m	0.11s ± 0.00 s
Conv-4	116.14 s ± 20.61 s	2.51 s ± 0.06s
ResNet-4	114.25 s ± 18.16 s	2.14 s ± 0.05 s
contextual CNN	394.94 s ± 68.81 s	5.01 s ± 0.09 s
FDSSC	1108.81 s ± 264.61 s	19.89 s ± 0.16 s
SSDC-DenseNet	608.28 s ± 117.93 s	9.16 s ± 0.43 s

(b)

Method	Training Time	Testing time
SAE-LR	505.74 m ± 6.63 m	0.07 s ± 0.00 s
Conv-4	184.59 s ± 36.00 s	4.95 s ± 0.02 s
ResNet-4	201.05 s ± 17.48 s	4.87 s ± 0.08 s
contextual CNN	844.43 s ± 67.89 s	11.25 s ± 0.02 s
FDSSC	3412.29 s ± 424.87 s	46.84 s ± 0.26 s
SSDC-DenseNet	829.02 s ± 171.96 s	11.01 s ± 1.27 s

(c)

the performance of the proposed SSDC-DenseNet model, several state-of-the-art DL-based methods, SAE-LR [22], Conv-4 [24], ResNet-4 [24], contextual CNN [27] and an FDSSC framework [28], are compared. The code of SAE-LR was offered by its original contributors and other methods were carefully developed by the authors according to the original papers. We also discuss the influence of different network architectures.

A. DATASET DESCRIPTION

We evaluated the performance of the proposed network on three publicly available HSI datasets: the Indian Pines (IN) dataset, University of Pavia (UP) dataset and Salinas dataset. The numbers of samples of each class in each dataset are shown in Table 1, Table 2 and Table 3, respectively. By testing the aforementioned methods with various datasets, we were able to explore the generalization ability of these models and provide convincing results.

The IN dataset was gathered by the airborne visible/infrared imaging spectrometer (AVIRIS) in 1992 from Northwest Indiana, and includes 16 vegetation classes. It has 145 × 145 pixels, with a resolution of 20 m by pixel. It originally had 220 bands, but 20 bands that were corrupted by water absorption effects were discarded. Hence, we used the remaining 200 bands ranging from 400 to 2500 nm to test the aforementioned methods.

TABLE 7. Number of parameters of different methods based on CNN.

Method	IN	UP	Salinas
Conv-4	101.3 k	62.4 k	102.5 k
ResNet-4	76.8 k	48.6 k	77.9 k
contextual CNN	1161.6 k	683.1 k	1892.2 k
FDSSC	1230.1 k	653.7 k	1254.1 k
SSDC-DenseNet	329.7 k	209.8 k	132.9 k

The UP dataset contains nine classes and consists of 610 × 340 pixels, with 103 spectral bands ranging from 0.43 to 0.86 μm. It has a spatial resolution of 1.3 m.

The Salinas dataset is characterized by a high spatial resolution of 3.7 m, and has 512 × 217 pixels with 224 spectral bands. Similar to the IN dataset, 20 water absorption bands were discarded. The dataset contains 16 classes.

B. EXPERIMENTAL SETUP

We tested our network with patch sizes $\{5 \times 5, 7 \times 7, 9 \times 9, 11 \times 11\}$ and found that the patch size of 7×7 was the most suitable to yield satisfactory performance. In the training process, with the batch size of 32, we adopted the Adam [34] optimizer to minimize the cross-entropy loss function. The initial learning rate was set to 0.0003, and reduced to 1/10 after 200 epochs. After another 100 epochs, it was reduced to 1/100 of the original learning rate.

All the training and testing results were obtained on the same computer, with the configuration of 16 GB of memory, NVIDIA GeForce GTX 1060 3GB and Intel i7 7700.

In our experiments, overall accuracy (OA), average accuracy (AA) and the Kappa coefficient (K) were chosen to evaluate the classification performance of the methods. To provide a statistical evaluation, each experiment was repeated 10 times, and the mean and standard deviation were reported.

We used two different ratios for ours and other compared methods. Each time, the training, validation and testing data was divided randomly according to the predetermined ratio. The splitting ratios for the training, validation and testing datasets were 15%:10%:75% for the IN dataset and 5%:15%:80% for both the UP and Salinas datasets, to demonstrate the classification performance.

C. COMPARISON WITH OTHER METHODS

To make sure every method could show its best performance, a suitable input data size should be carefully selected. We tested our model with different input spatial size and find that 7×7 could be the best for all datasets. Detailed information could be seen in Section III.D. For FDSSC, the experimental results show that the performance reached the peak when the patch size was 7×7, and then began to fall, which agrees with the trends illustrated in [28]. Hence, the input data for all the experiments on the three datasets were $7 \times 7 \times b$ volumes, where b is the number of bands. For all methods, except SAE-LR, the division of training, valida-

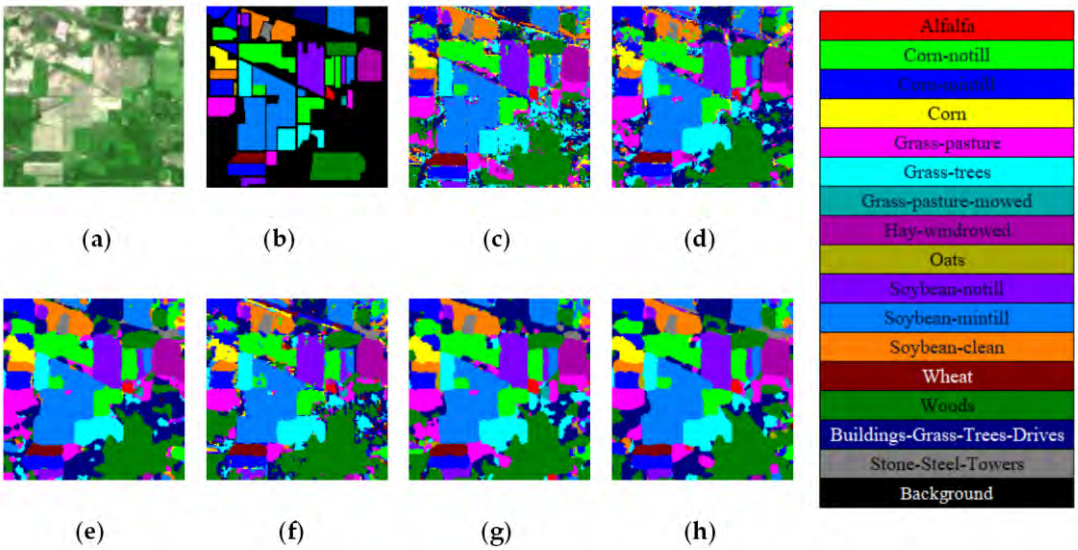

FIGURE 3. IN dataset: (a) false-color image, (b) ground truth image, and classification maps of (c) SAE-LR, (d) Conv-4, (e) ResNet-4, (f) contextual CNN, (g) FDSSC and (h) SSDC-DenseNet.

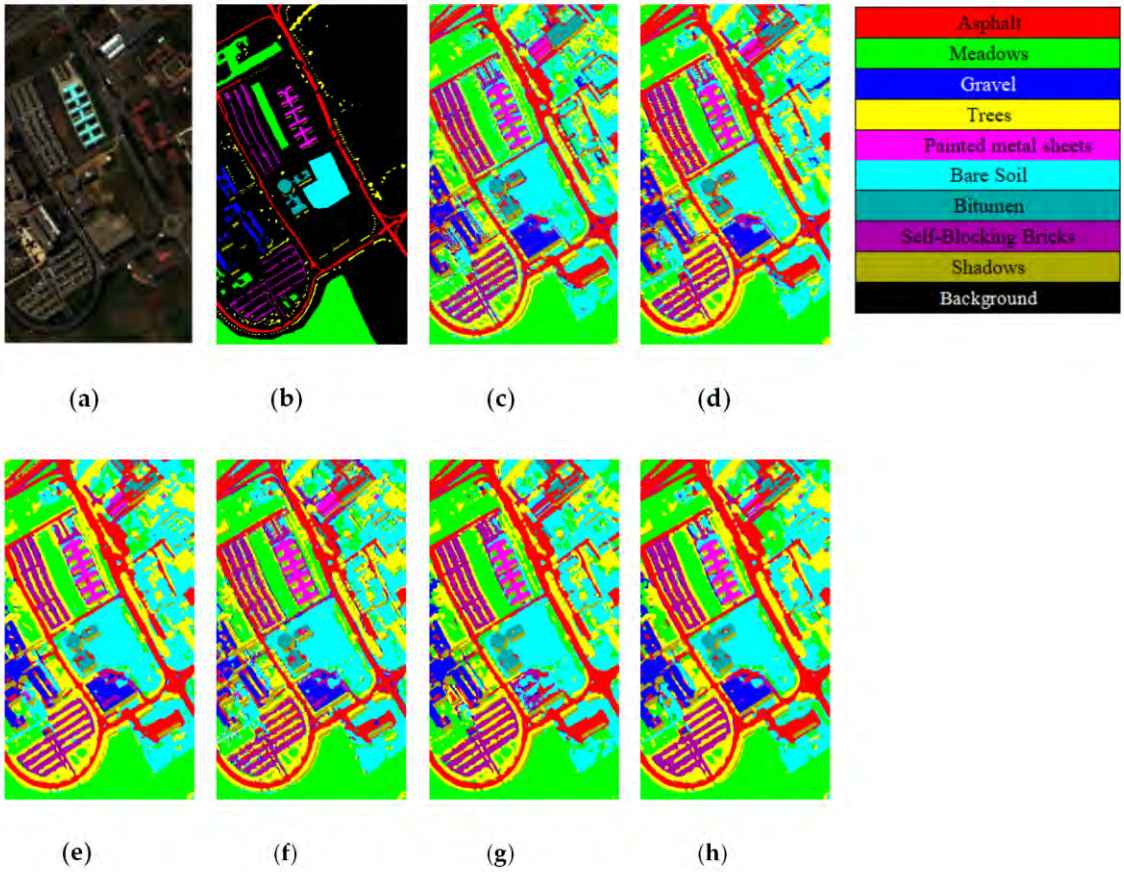

FIGURE 4. UP dataset: (a) false-color image, (b) ground truth image, and classification maps of (c) SAE-LR, (d) Conv-4, (e) ResNet-4, (f) contextual CNN, (g) FDSSC and (h) SSDC-DenseNet.

tion and testing data was same as that for SSDC-DenseNet. SAE-LR was constructed and trained according to the recommendation in [22] using the splitting ratio of 6:2:2 for all three datasets.

For the IN dataset, we found that SAE-LR trained for the KSC dataset in [22] was the most suitable, whereas SAE-LR trained for the UP dataset in [22] was the best for the UP and Salinas datasets in our experiments. Hence, we set the

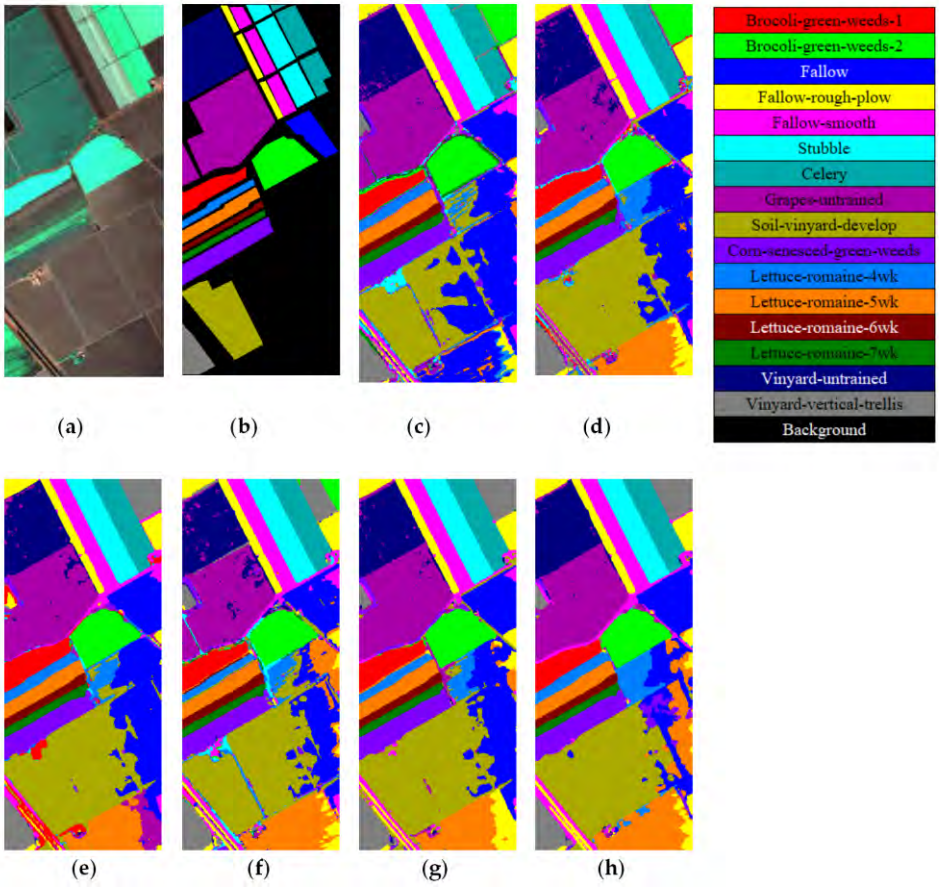

FIGURE 5. Salinas dataset: (a) false-color image, (b) ground truth image, and classification maps of (c) SAE-LR, (d) Conv-4, (e) ResNet-4, (f) contextual CNN, (g) FDSSC and (h) SSDC-DenseNet.

best configuration of SAE-LR for each dataset. Moreover, for all the models proposed in [24], Conv-4 and ResNet-4 achieved the best performance for the three datasets. Similarly, to obtain better performance, contextual CNN with 128 kernels in each layer was chosen for the IN dataset, and 192 kernels for the UP and Salinas datasets. After testing different configurations of SSDC-DenseNet in Section III.E, we finally chose three layers in each dense block with 48, 32 and 16 kernels per layer for the IN, UP and Salinas datasets, respectively.

In each experiment, the training data was selected randomly to demonstrate convincing results. For all methods, except SAE-LR, which is not CNN-based, we set the same batch size and learning rate for all the experiments for a fair comparison.

Table 4 and Fig. 3 show the results of the classification accuracies and visualized classification maps of the methods. The best performance in each set of experiments is indicated in bold font. Table 5 shows corresponding accuracies of the classification maps for each class in each dataset. We can see the superiority of our method in almost each category, which could be well reflected in the amount of noises in the classification maps. For the IN dataset, our network outperformed the other methods for OA, AA and K. Note that the proposed SSDC-DenseNet yielded 99.47% for OA, which was 5.35% higher than that of the SAE-LR method. For the other state-of-the-art techniques, our model obtained 5.75% and 0.84% higher results than 93.72% produced by CNN-4 and 98.63% by ResNet-4, respectively. It also demonstrated a slight improvement over other methods for all the metrics. It is worth noting that for large datasets, such as the UP and Salinas datasets, more recognizable improvements of our SSDC-DenseNet over FDSSC were demonstrated; the accuracies all reached 99%. Furthermore, SSDC-DenseNet had lower standard deviation, particularly in the case of the larger datasets.

Regarding the computational cost, the numbers of parameters and the training times required by the different methods are listed in Tables 6 and 7. Clearly, SAE-LR required the longest time for training. Although ResNet-4 consumed less computational cost, its test accuracies were below average. Contextual CNN required a large number of parameters to achieve its best performance, whereas the accuracy was also not the best. For the 3D network FDSSC, both the number of parameters, and times for training and testing were much higher than those of SSDC-DenseNet, which achieved better classification accuracy. The proposed SSDC-DenseNet had similar training and testing times to contextual CNN, and the

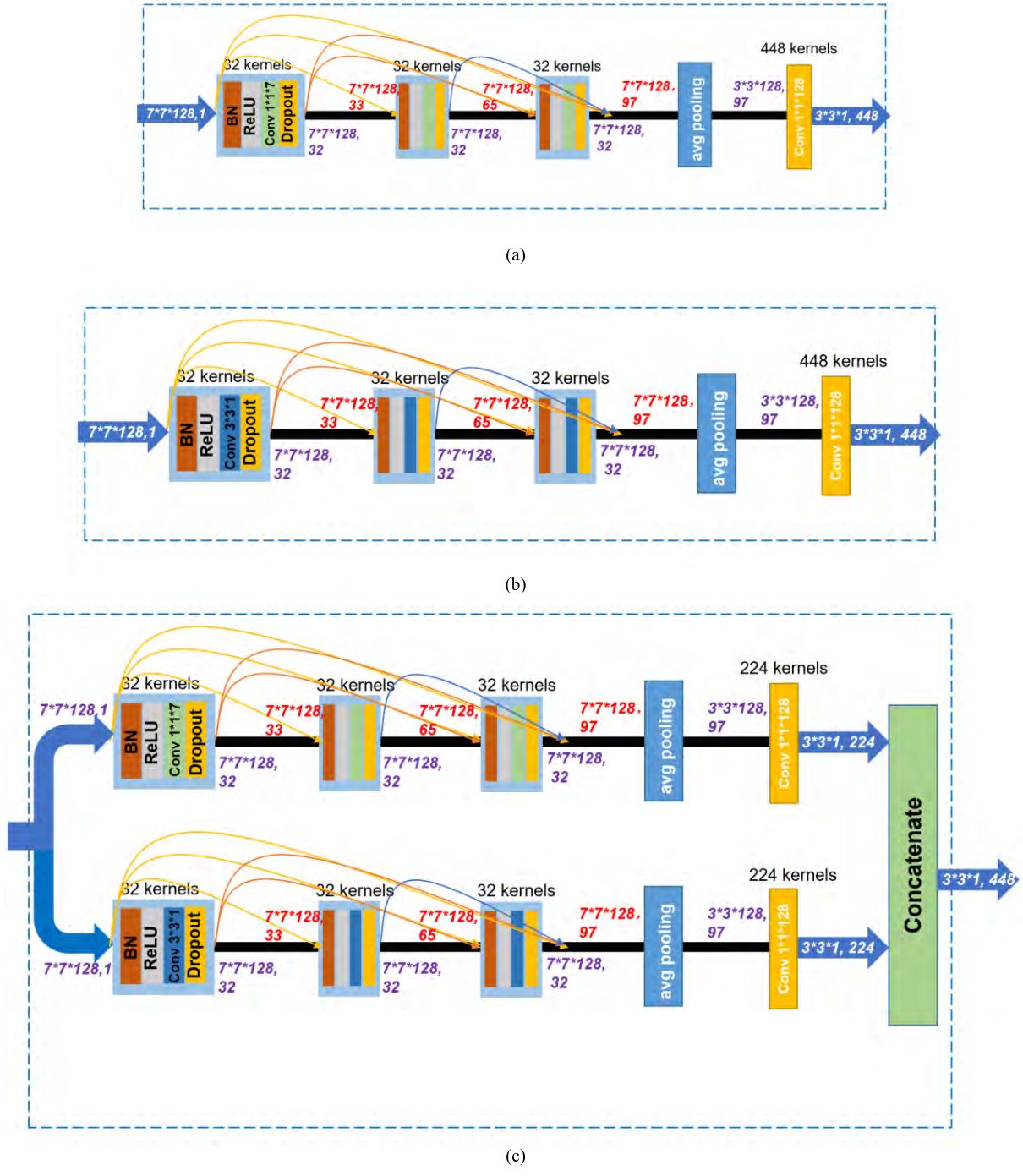

FIGURE 6. Structures of Block 2 (a)3D Spectral DenseNet (b)3D Spatial DenseNet (c)3D Spectral-Spatial DenseNet.

number of parameters was only 1/3 of those of contextual CNN.

D. COMPARISON ON THE COST BETWEEN THE 2D AND 3D DENSE MODULES

In this section, we replaced Block 2 with a spectral features learning dense module with $1 \times 1 \times 7$ filters (3D Spectral DenseNet), a spatial features learning dense module with $3 \times 3 \times 1$ filters (3D Spatial DenseNet) and a dual channel spectral-spatial features learning 3D dense module (3D Spectral-Spatial DenseNet) respectively. The details of the structures are given in Fig. 6. The flops, training time and OAs of different structures are given in Table 8. These experiments were conducted on the IN datasets with 15%:10%:75% splitting ratio. It can be noticed that although the 3D modules need more computational cost, the accuracy

TABLE 8. Comparison on the cost between the 2D and 3D dense modules.

	SSDC-DenseNet		3D Spectral DenseNet		3D Spatial DenseNet		3D Spectral-Spatial DenseNet	
	Total	Block2	Total	Block2	Total	Block2	Total	Block2
Flops($\times 10^7$)	58.68	48.44	1242.36	1232.11	1496.69	1486.44	2408.41	2398.16
Parameter sizes($\times 10^3$)	223.38	155.52	5652.60	5584.74	5658.93	5591.08	5681.31	5613.45
Training time (s)	484.11±65.16		2065.27±414.62		1925.97±211.74		3399.42±407.7	
OA(%)	99.47 ± 0.17		99.24±0.17		99.11±0.38		99.17±0.17	

TABLE 9. Training time(s) of SSDC-DenseNets with different kernel numbers per layer in Block 2.

	IN	UP	Salinas
16	439.50 ± 79.00	703.12 ± 125.43	829.02 ± 171.96
32	390.95 ± 60.34	608.28 ± 117.93	982.74 ± 143.41
48	484.11 ± 65.16	564.67 ± 128.31	982.74 ± 143.41
64	460.53 ± 121.23	660.19± 114.18	863.13 ± 158.45

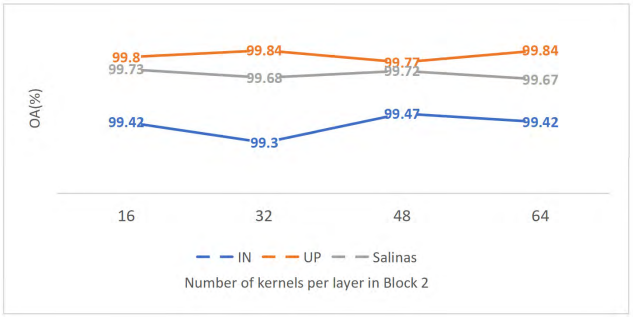

FIGURE 7. OA(%) of SSDC-DenseNets with different kernel numbers per layer in Block 2.

TABLE 10. Training time(s) of SSDC-DenseNets with different numbers of layers per dense block in Block 2.

	IN	UP	Salinas
3	390.95 ± 60.34	608.28 ± 117.93	863.13 ± 158.45
4	469.64 ± 188.21	755.79 ± 124.41	982.73 ± 154.59
5	486.13 ± 68.92	776.62 ± 188.21	1010.04 ± 140.23

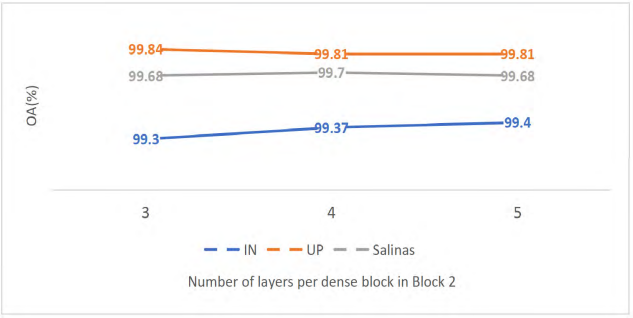

FIGURE 8. OA(%) of SSDC-DenseNets with different numbers of layers per dense block in Block 2.

improvement is quite limited. For example, the 3D Spectral DenseNet has twenty times larger parameter size and flops than that of the SSDC-DenseNet, but it only yields 99.24% OA.

It can be seen that networks with 3D dense modules need greater flops, computation power and concatenate memory requirement than our proposed SSDC-DenseNet to achieve competing HSI classification accuracy. We believe the cost effectiveness of the 2D dense modules lies in the simplicity of 2D convolution process.

E. DISCUSSION ON THE FRAMEWORK SETTINGS

In this section, we present the results of the proposed SSDC-DenseNet for a set of configurations. The experiments were designed mainly for two purposes: first, to search the optimal scale of the network, and second, to observe how the network reacts to changes of input information by adjusting the division of training, validation set and testing set, in addition to the size of input patches.

For the first purpose, we explored the effect of kernel numbers and layers in dense blocks in Block 2. The default configuration was three layers with 32 kernels per layer. When varying the framework settings, we kept the remaining factors unchanged.

First, we tested the SSDC-DenseNet with kernel numbers per layer of {16, 32, 48, 64}. Table 9 shows the training times and Fig. 7 displays the OAs for different configurations. With the kernel numbers of 16 and 32, the Salinas and UP datasets, respectively, achieved the best classification accuracy. When the kernel number was 48, performance peaked for the IN dataset.

We tested different layer numbers: {3, 4, 5}. The training time and OAs are shown in Table 10 and Fig. 8, respectively. We observe that the accuracy for the IN dataset kept growing as the layer number increased. For the UP dataset, the model with three layers achieved the highest accuracy. For the Salinas dataset, four layers was the proper choice.

From the results, we observe that the IN dataset tended to require a large model scale to achieve better accuracy, whereas the other two datasets were more suitable for fewer parameters. Moreover, as a trade-off between classification performance and computational cost, we chose the configuration of three layers with 48 kernels per layer for the IN dataset. For the UP and Salinas datasets, the models with three layers and kernel numbers of 32 and 16, respectively, were chosen. These models were applied in the following experiments to obtain the best performance for each dataset.

TABLE 11. Effect of the training fraction for the three datasets: (a) IN, (b) UP and (c) Salinas.

Training Fraction	10%	15%	20%	25%	30%
OA(%)	98.70 ± 0.31	99.47 ± 0.17	99.70 ± 0.10	99.79 ± 0.08	**99.81 ± 0.08**
AA(%)	98.31 ± 0.45	99.29 ± 0.29	99.58 ± 0.29	99.69 ± 0.19	**99.78 ± 0.14**
K × 100	98.51 ± 0.35	99.39 ± 0.20	99.66 ± 0.11	99.76 ± 0.09	**99.78 ± 0.09**

(a)

Training Fraction	5%	10%	15%	20%	25%
OA(%)	99.84 ± 0.05	99.95 ± 0.02	99.97 ± 0.02	99.98 ± 0.01	**99.99 ± 0.01**
AA(%)	99.74 ± 0.11	99.92 ± 0.03	99.40 ± 0.03	99.97 ± 0.02	**99.98 ± 0.02**
K × 100	99.79 ± 0.08	99.30 ± 0.03	99.60 ± 0.02	99.98 ± 0.02	**99.99 ± 0.01**

(b)

Training Fraction	5%	10%	15%	20%	25%
OA(%)	99.73 ± 0.07	99.92 ± 0.04	99.98 ± 0.01	99.99 ± 0.01	**99.99 ± 0.01**
AA(%)	99.82 ± 0.06	99.93 ± 0.03	99.98 ± 0.02	99.99 ± 0.01	**99.99 ± 0.01**
K × 100	99.70 ± 0.08	99.91 ± 0.05	99.98 ± 0.01	99.99 ± 0.01	**99.99 ± 0.01**

(c)

TABLE 12. OA(%) for the three datasets with varying input spatial sizes.

Dataset	5×5	7×7	9×9	11×11
IN	98.87 ± 0.33	**99.47 ± 0.17**	99.39 ± 0.20	99.33 ± 0.12
UP	99.56 ± 0.09	**99.84 ± 0.05**	99.79 ± 0.08	99.78 ± 0.05
Salinas	99.24 ± 0.12	99.73 ± 0.07	99.89 ± 0.06	**99.93 ± 0.04**

Additionally, we tested the SSDC-DenseNet with different fractions of training sets. As shown in Table 10, for the IN dataset, we tested the training fraction {10%, 15%, 20%, 25%, 30%} and testing fraction {80%, 75%, 75%, 70%, 65%}. Considering that the UP and Salinas datasets have more samples, the splitting ratio that we set for them was {5%:15%:80%, 10%:15%:75%, 15%:10%:75%, 20%:5%:75%, 25%:5%:70%}. Clearly, the proposed SSDC-DenseNet achieved good results with a small training fraction. As the training fraction increased, the classification accuracy tended to grow. Additionally, we note that our network demonstrated good robustness and generalizability with varying testing fractions.

From Table 11, it is obvious that the accuracy increases as the training set proportion becomes larger. Note that when the training fraction was larger than 15%, the classification accuracies reached 99% for the IN dataset, whereas for the other two datasets, the accuracies all reached 99.9%. Clearly, a large training set was not necessary for our SSDC-DenseNet. Furthermore, to balance the tradeoff between computational cost and performance, the 15% training fraction for the IN dataset and 5% for UP and Salinas datasets were reasonable settings.

With respect to the impact of the input patch size, SSDC-DenseNet demonstrated robustness toward varying numbers of input spatial sizes. We chose a set of patch sizes to assess our network:$5 \times 5 \times b$, $7 \times 7 \times b$, $9 \times 9 \times b$, $11 \times 11 \times b$, where b is the spectral number of the original image. In Table 12, we observe that when the spatial size became larger in a certain range, the network tended to yield a higher accuracy, particularly for the Salinas dataset, partly because the larger spatial size provided more abundant spatial information. When the spatial size was too large for the network, the accuracy began to decrease. We set the spatial size to $7 \times 7 \times b$ in our experiments.

IV. CONCLUSION

In this paper, to reduce the computational cost in spectral-spatial approaches while maintaining the high accuracy, we proposed a novel end-to-end SSDC-DenseNet. Extensive experiments were conducted based on three publicly available datasets. In particular, by validating the model with different framework configurations and varying training samples, we proved its robustness and generalization ability in HSI classification. The proposed SSDC-DenseNet demonstrated competitive performance compared with other DL-based methods: SAE-LR, CNN, ResNet, contextual CNN and FDSSC.

Furthermore, we observed that our model can exploit high-level features with the effective introduction of dense blocks. By jointly using spectral and spatial features with 2D multi-scale filters, it is possible for SSDC-DenseNet to yield satisfactory accuracy with fewer parameters, compared with other methods. Moreover, the reduction of training time could be clearly observed, particularly for large datasets. At the same time, the proposed network has much less parameters than various methods while keeping the high accuracy. The above advantages enable our network to gain high accuracy with relatively low computational cost in a spectral-spatial way.

ACKNOWLEDGMENT

The authors thank Maxine Garcia, PhD, from Edanz Group (www.edanzediting.com/ac) for editing a draft of the manuscript.

REFERENCES

[1] D. Manolakis, D. Marden, and G. A. Shaw, "Hyperspectral image processing for automatic target detection applications," *Lincoln Lab. J.*, vol. 14, no. 1, pp. 79–116, 2003.

[2] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. M. Nasrabadi, and J. Chanussot, "Hyperspectral remote sensing data analysis and future challenges," *IEEE Geosci. Remote Sens. Mag.*, vol. 1, no. 2, pp. 6–36, Jun. 2013.

[3] P. C. Mohanty, S. Panditrao, R. S. Mahendra, H. S. Kumar, and T. S. Kumar, "Identification of coral reef feature using hyperspectral remote sensing," *Proc. SPIE*, vol. 9880, Apr. 2016, Art. no. 98801B.

[4] Ø. Ødegård, A. A. Mogstad, G. Johnsen, A. J. Sørensen, and M. Ludvigsen, "Underwater hyperspectral imaging: A new tool for marine archaeology ," *Appl. Opt.*, vol. 57, no. 12, pp. 3214–3223, 2018. doi: 10.1364/AO.57.003214.

[5] M. Fauvel, Y. Tarabalka, J. A. Benediktsson, J. Chanussot, and J. C. Tilton, "Advances in spectral-spatial classification of hyperspectral images," *Proc. IEEE*, vol. 101, no. 3, pp. 652–675, Mar. 2013. doi: 10.1109/JPROC.2012.2197589.

[6] D. Casasent and X.-W. Chen, "Feature reduction and morphological processing for hyperspectral image data," *Appl. Opt.*, vol. 43, no. 2, pp. 227–236, 2004. doi: 10.1364/AO.43.000227.

[7] H. Su, Y. Sheng, P. Du, and K. Liu, "Adaptive affinity propagation with spectral angle mapper for semi-supervised hyperspectral band selection," *Appl. Opt.*, vol. 51, no. 14, pp. 2656–2663, 2012. doi: 10.1364/AO.51.002656.

[8] D. A. Landgrebe, *Signal Theory Methods in Multispectral Remote Sensing*. Hoboken, NJ, USA: Wiley, 2005.

[9] J. Xia, J. Chanussot, P. Du, and X. He, "(Semi-) supervised probabilistic principal component analysis for hyperspectral remote sensing image classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 6, pp. 2224–2236, Jun. 2014. doi: 10.1109/jstars.2013.2279693.

[10] J. Wang and C.-I. Chang, "Independent component analysis-based dimensionality reduction with applications in hyperspectral image analysis," *IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 6, pp. 1586–1600, Jun. 2006. doi: 10.1109/TGRS.2005.863297.

[11] C. Xu, C. Lu, J. Gao, W. Zheng, T. Wang, and S. Yan, "Discriminative analysis for symmetric positive definite matrices on Lie groups," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 10, pp. 1576–1585, Oct. 2015. doi: 10.1109/TCSVT.2015.2392472.

[12] M. Fauvel, J. A. Benediktsson, J. Chanussot, and J. R. Sveinsson, "Spectral and spatial classification of hyperspectral data using SVMs and morphological profiles," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 11, pp. 3804–3814, Nov. 2008. doi: 10.1109/TGRS.2008.922034.

[13] K. Bakos and P. Gamba, "Hierarchical hybrid decision tree fusion of multiple hyperspectral data processing chains," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 1, pp. 388–394, Jan. 2011. doi: 10.1109/TGRS.2010.2051554.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. 26th Annu. Conf. Neural Inf. Process. Syst.*, Lake Tahoe, NV, USA, vol. 2, Dec. 2012, pp. 1097–1105.

[15] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017. doi: 10.1109/TPAMI.2016.2577031.

[16] A. O. B. Özdemir, B. E. Gedik, and C. Y. Y. Çetin, "Hyperspectral classification using stacked autoencoders with deep learning," in *Proc. 6th Workshop Hyperspectral Image Signal Process., Evol. Remote Sens. (WHISPERS)*, Lausanne, Switzerland, Jun. 2014, pp. 1–4. doi: 10.1109/WHISPERS.2014.8077532.

[17] Y. Chen, X. Zhao, and X. Jia, "Spectral–spatial classification of hyperspectral data based on deep belief network," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 8, no. 6, pp. 2381–2392, Jun. 2015. doi: 10.1109/JSTARS.2015.2388577.

[18] J. Li, B. Xi, Y. Li, Q. Du, and K. Wang, "Hyperspectral classification based on texture feature enhancement and deep belief networks," *Remote Sens.*, vol. 10, no. 3, p. 396, Mar. 2018, Art no. 396. doi: 10.3390/rs10030396.

[19] W. Hu, Y. Huang, L. Wei, F. Zhang, and H. Li, "Deep convolutional neural networks for hyperspectral image classification," *J. Sensors*, vol. 2015, Jan. 2015, Art. no. 258619. doi: 10.1155/2015/258619.

[20] V. Slavkovikj, S. Verstockt, W. De Neve, S. Van Hoecke, and R. Van de Walle, "Hyperspectral image classification with convolutional neural networks," in *Proc. 23rd ACM Int. Conf. Multimedia*, Oct. 2015, pp. 1159–1162.

[21] H. Liang and Q. Li, "Hyperspectral imagery classification using sparse representations of convolutional neural network features," *Remote Sens.*, vol. 8, no. 2, p. 99, Feb. 2016. doi: 10.3390/rs8020099.

[22] Y. Chen, Z. Lin, X. Zhao, G. Wang, and Y. Gu, "Deep learning-based classification of hyperspectral data," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 6, pp. 2094–2107, Jun. 2014. doi: 10.1109/JSTARS.2014.2329330.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[24] Z. Zhong, J. Li, L. Ma, H. Jiang, and H. Zhao, "Deep residual networks for hyperspectral image classification," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Fort Worth, TX, USA, Jul. 2017, pp. 1824–1827.

[25] J. Yang, Y. Zhao, J. C.-W. Chan, and C. Yi, "Hyperspectral image classification using two-channel deep convolutional neural network," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Beijing, China, Jul. 2016, pp. 5079–5082.

[26] Y. Chen, H. Jiang, C. Li, X. Jia, and G. Pedram, "Deep feature extraction and classification of hyperspectral images based on convolutional neural networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 10, pp. 6232–6251, Oct. 2016. doi: 10.1109/TGRS.2016.2584107.

[27] H. Lee and H. Kwon, "Going deeper with contextual CNN for hyperspectral image classification," *IEEE Trans. Image. Process.*, vol. 26, no. 10, pp. 4843–4855, Oct. 2017. doi: 10.1109/TIP.2017.2725580.

[28] W. Wang, S. Dou, Z. Jiang, and L. Sun, "A fast dense spectral–spatial convolution network framework for hyperspectral images classification," *Remote Sens.*, vol. 10, no. 7, p. 1068, Jul. 2018. doi: 10.3390/rs10071068.

[29] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, College Park, MD, USA, Jul. 2017, pp. 1–9.

[30] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, Fort Lauderdale, FL, USA, vol. 15, Apr. 2011, pp. 315–323.

[31] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 1–9.

[32] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. 32nd Int. Conf. Mach. Learn. (ICML)*, Lile, France, vol. 1, Jul. 2015, pp. 448–456.

[33] M. Lin, Q. Chen, and S. Yan, "Network in network," *CoRR*, vol. abs/1312.4400, pp. 1–10, Dec. 2013.

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, May 2015, pp. 7–9.

YUTONG BAI is currently pursuing the B.S. degree in software engineering with Sichuan University, Chengdu, China. Her research interests include deep learning, remote sensing, and image processing.

QIFAN ZHANG is currently pursuing the B.S. degree in software engineering with Sichuan University, Chengdu, China. His research interests include deep learning, machine learning, and image processing.

ZEXIN LU received the B.S. degree in computer science from the Beijing University of Chemical Technology, Beijing, China, in 2018. He is currently pursuing the M.S. degree in software engineering with Sichuan University, Chengdu, China. His research interests include X-ray computed tomography, machine learning, and deep learning.

YI ZHANG (S'11–M'12–SM'18) received the B.S., M.S., and Ph.D. degrees from the College of Computer Science, Sichuan University, Chengdu, China, in 2005, 2008, and 2012, respectively. From 2014 to 2015, he was with the Department of Biomedical Engineering, Rensselaer Polytechnic Institute, Troy, NY, USA, as a Postdoctoral Researcher. He is currently an Associate Professor with the College of Computer Science, Sichuan University. His research interests include medical imaging, remote sensing, compressive sensing, and deep learning. He is currently an Associate Editor of IEEE Access.