Hyperspectral Remote Sensing Image Classification Using Deep Convolutional Capsule Network

Runmin Lei[1], Chunju Zhang[1,2,4], Wencong Liu[1], Lei Zhang[1], Xueying Zhang[3], Yucheng Yang[1], Jianwei Huang[1], Zhenxuan Li[1] and Zhiyi Zhou[1]

1 [1]School of Civil Engineering, Hefei University of Technology, Hefei, 230009, China

[2]Key Laboratory of Urban Land Resources Monitoring and Simulation, Ministry of Natural Resources, Shenzhen, 518000, China

[3]Key Laboratory of virtual geographic environment, Nanjing Normal University, Nanjing, 210023, China

[4]Intelligent Interconnected Systems Laboratory of Anhui Province (Hefei University of Technology), Hefei, 230009, China

Abstract— **Deep learning models have shown excellent performance in the hyperspectral remote sensing image (HSI) classification. In particular, convolutional neural networks (CNNs) have received widespread attention because of their powerful feature-extraction ability. Recently, a capsule network (CapsNet) was introduced to boost the performance of CNNs, marking a remarkable progress in the field of HSI classification. In this paper, we propose a novel deep convolutional capsule neural network (DC-CapsNet) based on spectral–spatial features to improve the performance of CapsNet in the HSI classification, while significantly reducing the computation cost of the model. Specifically, a convolutional capsule layer based on the extension of dynamic routing using 3D convolution is used to reduce the number of parameters and enhance the robustness of the learned spectral–spatial features. Furthermore, a lighter and stronger decoder network composed of deconvolutional layers as a better regularization term and capable of acquiring more spatial relationships is used to further improve the HSI classification accuracy with low computation cost. In this study, we tested the performance of the proposed model on four widely used HSI datasets: the Kennedy Space Center, Indian Pines, Pavia University and Salinas datasets. We found that the DC-CapsNet achieved high classification accuracy with limited training samples and effectively reduced the computation cost.**

Index Terms— **capsule neural network, convolutional neural network, hyperspectral image classification**

I. Introduction

HYPERSPECTRAL remote sensing images have been widely used in various fields because of their rich spectral and spatial information; these fields include precision agriculture [1], land change monitoring [2], mineral exploitation [3], and scene recognition [4]. Hyperspectral remote sensing image (HSI) classification has become a hot topic in the field of remote sensing in recent years [5–8]. In the [1]past few decades, a large number of methods have been proposed for HSI classification. There are some traditional methods based on spectral information used to classify HSIs, such as k-nearest neighbor (k-NN) [9], extreme learning machine (ELM) [10], support vector machine (SVM) [11], and wavelet transform [12]. However, due to only using the spectral information of pixels for classification, they do not have good noise robustness. Therefore, some methods based on spectral–spatial information have been proposed to improve the accuracy of HSI classification [13], which can fully mine and utilize the dependence between local pixels within HSIs. In a previous study [14], a 3D Gabor-wavelet-based method that implemented a set of complex Gabor wavelets to extract spectral–spatial features from HSIs was used. With the improvement of SVMs, the morphological profiles and the Markov random field were introduced to advance the HSI classification accuracy [15,16]. Nevertheless, these methods can only extract the shallow features of the input HSIs. Typically, HSI data have a high dimension and highly nonlinear structure. Moreover, the spatial and spectral resolution of HSIs have been improved with the development of sensor technology, resulting in the ground objects have more obvious details and more complex spectral characteristics. The phenomenon of different objects having the same spectrum and the same objects having different spectra is becoming interesting due to more and more abundant spectral information. In this instance, it is difficult to obtain a good classification result using only the shallow features of HSIs. Deep learning methods can automatically extract abstract features from low to high levels in an end-to-end manner, which can learn more robust features and provide an accurate HSI classification result. Some typical deep learning models have been used for HSI classification, including stacked autoencoder (SAE) [17], deep belief network (DBN) [18], recurrent neural network (RNN) [19], and convolutional neural networks (CNNs) [20–22], which greatly improved the accuracy of HSI classification. Owing to the inherent characteristics of weight sharing and local connection, CNN has attracted much attention in the field of image processing. For instance, a 1D convolutional neural network (1D-CNN) was proposed to accurately classify HSIs by extracting spectral features [23]. However, the 1D-CNN takes the feature vector of the spectral signal as the input of the model, while ignoring many spatial information of HSIs. To consider the rich spatial information in HSIs, 2D convolutional neural networks (2D-CNNs) have been proposed to extract spatial and spectral information and reduce the dimension of

[1] This work was supported by the Open Fund of Key Laboratory of Urban Land Resources Monitoring and Simulation, Ministry of Natural Resources (KF-2020-05-084), and the National Natural Science Foundation of China (41971337), and the Fundamental Research Funds for the Central Universities of China (JZ2021HGTB0088, JZ2021HGTA0169 and JZ2021HGTA0167). (Corresponding author: Chunju Zhang).

the spectral domain of the origin HSI using descending dimension algorithms [24–26]. Nevertheless, these methods usually lead to the spatial information lost during the preprocessing stage. To make full use of the spectral-spatial information in HSIs, 3D convolutional neural networks (3D-CNNs) [27,28] have been presented to simultaneously extract the spectral–spatial features from the original HSI data and obtain better classification accuracy. Generally, the deeper features are more discriminative, but repeating convolution layers causes optimization difficulties. To solve this problem, residual learning and dense connectivity have been introduced into the CNN to form a spectral–spatial residual network (SSRN) [29] and a deep&dense CNN [30], which can obtain deeper networks and provide a good performance. Furthermore, some powerful techniques have been combined with CNNs to enhance the performance of HSI classification. Zhang *et al.* [31,32] and Mu *et al.* [33] introduced feature fusion techniques to combine different scale information to enhance the robustness of deep networks against overfitting. Liu *et al.* [34] performed transfer learning between different HSI datasets to improve the HSI classification for small-sample conditions. To learn more representative features from the original HSI, sparse representation [35], metric learning [36], and attention techniques [37] are also used for refining the learned spectral–spatial features. Moreover, some learning-based methods introduced the generative adversarial network (GAN) to improve the classification accuracy and mitigate the overfitting risk [38,39].

Although deep learning models, especially CNN-based models, have achieved considerable progress in the field of HSI classification, there are still some disadvantages limiting the performance of the model. Firstly, CNN-based methods have a complex network structure and require a large number of labeled samples to train the model. However, the number of labeled samples is limited which is a common bottleneck in the HSI classification field. Secondly, CNNs usually use max pooling operations to reduce the computation cost and advance the invariance of features which can capture more discriminative features but lose the relationship between the features of geographic objects. Thus, it can only learn shallow spatial features, while ignoring the important knowledge of spatial relationships and patterns for HSIs. Thirdly, owing to the complexity of HSIs, the scalar value used to represent features in CNNs shows poor representation ability. A novel type of deep learning model called capsule network (CapsNet) [40], which uses dynamic routing-by-agreement and vector-output capsules to encode the relationships between different features and enhance the feature representation ability of the model has been previously proposed to improve the performance of CNNs. The length of the activity vector output by the capsule represents the estimated probability that the target object exists in the input image, and the orientation of the vector indicates the capsule properties. It can effectively dig deep information of spatial knowledge implied in HSIs, namely, spatial relationships and patterns, which plays an important factor in the high-level cognition of the ground object. It is true that CapsNet can significantly improve the performance of deep learning models in the HSI classification [41-43]. In a previous study [44], a spectral–spatial capsule network is proposed to accurately classify HSIs. Unfortunately, owing to the fully connected method in dynamic routing, the model has a large number of trainable parameters and massive computation cost. It is easily overfitted with insufficient training samples, which have a negative influence on the HSI classification performance of the model. Therefore, Yin *et al.* [45] used transfer learning to initialize the convolutional parameters of CapsNet and dramatically enhance the classification accuracy. Lei *et al.* [46] proposed a non-local CapsNet combined attention technique and CapsNet to capture non-local spectral-spatial features. Li *et al.* [47] introduced the maximum correntropy criterion (MCC) to generate a robust 3-D-CapsNet. Another way to address the aforementioned problems is to conduct architecture improvements. Zhang *et al.* [48] and Zhu *et al.* [49] introduced a convolutional capsule layer based on local connections and shared transform matrices during dynamic routing to address the lack of labeled samples and ensure high precision. Nevertheless, simply shared transform matrices in the local receptive field could not generate the appropriate prediction vector for each child capsule, leading to a limited performance of the HSI classification model. Furthermore, the complex preprocessing will damage the spectral information of HSIs, and a large spatial size of input data generally leads to the training and test samples being very overlapped, which could increase the additional calculation cost [48,49].

In this study, we developed a novel deep convolutional capsule network (DC-CapsNet) for effective HSI classifications, which is based on 3D convolutional capsule layers. In DC-CapsNet, an extension of dynamic routing inspired by 3D convolution [50] is used to reduce the number of parameters and suppress the adverse effect of stacking capsule layers. It will be able to capture more robust higher-level feature information, enabling the model to provide accurate classification results with limited training samples and a low computation cost. Moreover, the reconstruction loss, which is generated by the decoder network, consists of several fully connected layers used in [40] and shows great potential for reducing overfitting and improving the HSI classification performance of CapsNet-based models [44-47]. However, the decoder network composed of the fully connected layers has too many parameters, which lead to massive computational consumption during the training stage of the model. Moreover, the fully connected layer makes it difficult to capture spatial information, and the introduction of additional layers implicitly increases the complexity of the model. Therefore, in this study, a lighter and stronger decoder network composed of deconvolutional layers is used as a better regularization term that can acquire more spatial relationships to further improve the HSI classification accuracy and lower the computation cost.

The major contributions of this paper are listed as follows.

1) We propose a novel deep convolutional capsule network, which can dig deep information of spatial and spectral knowledge implied in HSIs, especially, spatial relationships and patterns of the ground object; a novel dynamic routing

based on 3D convolution is used to boost the classification performance and reduce the trainable parameters of DC-CapsNet.

2) A lighter and stronger decoder network consisting of deconvolutional layers is proposed to further improve the classification accuracy of HSIs and lower the computation cost.

3) The proposed model is evaluated on four well-known HSI datasets and provides promising classification results with limited training samples. Moreover, the DC-CapsNet can effectively reduce the computation cost.

The remainder of this paper is organized as follows: Section II describes the detailed architecture of the DC-CapsNet, Section III presents the results of the experiments and discussion, and Section IV concludes the paper.

II. Proposed Framework

Inspired by the success achieved by CNNs (by going deeper) and the fact that the deep features are generally more discriminative, in this study, we considered a deeper capsule network based on CapsNet with more layers to solve the problem of limited labeled samples in the HSI classification. In this section, we explain the specific details of the DC-CapsNet.

A. 3D convolutional capsule layer

CapsNet [40] is a novel neural network architecture that generally has a simpler structure than CNN-based models used for HSI classification. It uses a capsule encapsulated by a group of neurons as the base unit, and each capsule outputs an activity vector rather than a scalar value to represent a specific type of entity. The length of the vector indicates the probability that the geographic entity exists in the input HSI image, whereas the orientation denotes the properties of the geographic entity, such as translation, rotation, and scale. It shows a more powerful representation ability for the complex surface environment and higher dimensionality of HSIs. Furthermore, CapsNet replaces max pooling with dynamic routing to capture the spatial relationship between different features, which is important for HSI classification. Compared with max pooling, using dynamic routing allows the model to independently adjust the strength of the connection between the child capsule and the potential parent capsules through iteration. It is much more effective than max pooling in improving the invariance of the extracted HSI features.

However, the original capsule layer is fully connected and contains a large number of trainable parameters. Simply stacking additional capsule layers can compromise the performance of the model and result in computationally inefficiency [51]. In particular, it is easy to overfit because of the limited number of training samples in the HSI classification. Thus, the local connection and capsule layer are combined to form a 3D convolutional capsule layer, which could significantly reduce the trainable parameters and computational cost as well as enhance the representation capacity of activity vectors output by using capsules. Thus, it can allow the model to provide accurate classification results in the case of limited training samples.

To construct a robust deep convolutional capsule network for HSI classification, an extension of dynamic routing based on 3D convolution [50] was used to boost the performance of DC-CapsNet (Fig. 1). The output of capsule layer l, denoted as $\mathbf{\Phi}^l \in \mathbf{R}^{(w^l, w^l, c^l, n^l)}$, was composed of c^lcapsule tensors, which in turn were composed of $w^l \times w^l$—a spectral–spatial capsule with n^l dimensions, where $w^l \times w^l$was the spatial size of the feature maps. First, $\mathbf{\Phi}^l$ was reshaped to a 3D tensor $\widetilde{\mathbf{\Phi}}^l$ with a shape of $\widetilde{\mathbf{\Phi}}^l$ and convolved by $c^{l+1} \times n^{l+1}$—the number of 3D convolution kernels. To maintain the integrity of the information output by the capsule, both the depthwise stride and depthwise kernel size of the 3D convolution were set to n^l. Then, the prediction $\mathbf{U}$ was be obtained using a reshape operation, having the shape of $(w^{l+1}, w^{l+1}, n^{l+1}, c^{l+1}, c^l)$.

$$\mathbf{U} = \text{Reshape}\ (\text{Conv3D}(\text{Reshape}(\mathbf{\Phi}^l))) \tag{1}$$

Generally, the convolutional operations are the foundation of the capsule layers, which can result in adjacent capsules sharing similar information. This means that a specific type of geographic entity is represented by an adjacent set of capsules. Using a 3D convolutional operation with a height and width kernel to generate prediction vectors allows us to predict a parent capsule using several adjacent sets of child capsules and focus on detailed information. This could help us to obtain more robust spectral–spatial features from HSIs. For example, we used 8 capsule sets, in which each set contains an adjacent 3 ×3 child capsule, to predict a parent capsule in this study. Therefore, the capsules were integrated into groups for routing instead of individual routing.

As shown in Fig. 1, each capsule tensor $s \in c^l$ in layer l generates a prediction $\mathbf{U}_s$, which consists of c^{l+1} the number of capsule tensors. In other words, the capsule in any position (p, q, r) of layer l+1 is related to all capsule tensors in layer l, where $p \in w^{l+1}, q \in w^{l+1}$, and $r \in c^{l+1}$. Thus, the coupling coefficients of the prediction $\mathbf{U}_s$ ($\mathbf{K}_s \in \mathbf{R}^{(w^{l+1}, w^{l+1}, c^{l+1})}$) can be obtained using a 3D version softmax function [50] for all s:

$$\mathbf{K}_s = \text{softmax}\ (\mathbf{B}_s), \tag{2}$$

$$k_{pqrs} = \frac{\exp\ (b_{pqrs})}{\sum_x \sum_y \sum_z \exp\ (b_{xyzs})}, \tag{3}$$

where the logits of the prediction $\mathbf{U}_s$ ($\mathbf{B}_s \in \mathbf{R}^{(w^{l+1}, w^{l+1}, c^{l+1})}$) are initialized to zero at the beginning and updated by the iterative dynamic routing process. k_{pqrs} and b_{pqrs} indicate the coupling coefficient and logit that in the position (p, q, r) of $\mathbf{K}_s$ and $\mathbf{B}_s$, respectively. In addition, the sum of all k_{pqrs} is 1 for all s.

Then, the total input to the capsule in layer l+1 (S_{pqr}) is a weighted sum of all prediction vectors U_{pqrs}. The final output V_{pqr} can be obtained through a squash function [40], which is used to ensure that the length of the capsule vector is between 0 and 1.

$$S_{pqr} = \sum_s k_{pqrs} \cdot U_{pqrs}, \tag{4}$$

$$V_{pqr} = \frac{\|S_{pqr}\|^2}{1 + \|S_{pqr}\|^2} \cdot \frac{S_{pqr}}{\|S_{pqr}\|}, \tag{5}$$

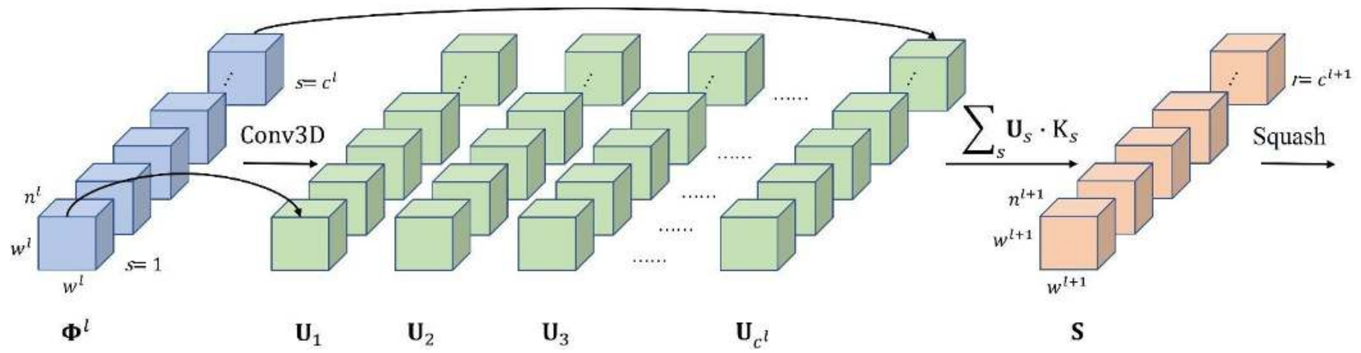


Fig. 1. Dynamic routing based on 3D convolution [50]

where V_{pqr} represents the capsule output in the position *(p, q, r)* of layer *l*+1, and the logit b_{pqrs} updated by measuring the agreement between V_{pqr} and U_{pqrs}.

$$b_{pqrs} \leftarrow b_{pqrs} + V_{pqr} \cdot U_{pqrs}, \tag{6}$$

The output of capsule layer *l*+1, denoted as $\mathbf{\Phi}^{l+1} \in \mathbf{R}^{(w^{l+1}, w^{l+1}, c^{l+1}, n^{l+1})}$, can be obtained using V_{pqr}. This process not only extracts the spectral–spatial features from the input feature maps but also models the part–whole data relationships of the target geographic object.

B. Encoder Network

The main framework of the encoder network is shown in Fig. 2. The input data of the encoder network is a 3D cube with a size of $w \times w \times L$ from the original HSI. Here, w is the height and width of the input image, and L indicates the spectral dimension of the input data. Because of the complexity of HSI data, two traditional convolutional layers were introduced in shallow layers to capture robust higher-level spectral–spatial features from the input HSI data; this was fed to the capsule layer. The third layer, namely, the PrimaryCaps layer, was the first capsule layer which converted the neurons into capsules. Its foundation is the traditional convolutional operation. This layer outputs a 4D tensor composed of c^1 feature maps, with a spatial size of $w^1 \times w^1$, where the dimension of each feature map is n^1 (the number of neurons for each capsule). The fourth layer, namely, the 3D convolutional capsule layer, is the core of the DC-CapsNet. Unlike simply shared transform matrices in the local receptive field, dynamic routing based on 3D convolution was adopted to obtain more robust higher-level capsules in this layer. Its output consists of c^2 feature maps, with n^2 dimensions and the spatial size of each feature map being $w^2 \times w^2$. The last layer was a fully connected capsule layer, with an output of *n_class* (the number of classes) capsule, and each capsule yielded a vector to represent a specific class of geographic entity. The length of the vector indicates the probability that the center pixel of the input data belongs to one class. Furthermore, the output vectors of these capsule layers provide the details about the target geographic object or part of the object of interest for HSI classification, including pose, orientation, and scale. The dynamic routing allows the model to capture the spatial relationships and patterns between features that are beneficial for accurate HSI classification.

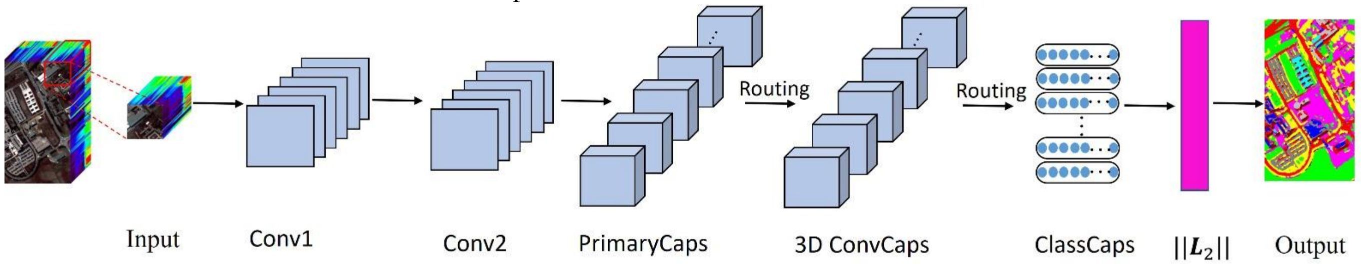


Fig. 2. Encoder network

In this paper, the margin loss [40] was used to calculate the difference between the output of the network and the expected output. It could enhance the length of the vector output by the top-level capsule corresponding to the target object, while suppressing the length of other vectors when that object is present in the input cube.

$$L_j = T_j \max(0, m^+ - \|v_j\|)^2 + \lambda\,(1 - T_j)\,\max(0, \|v_j\| - m^-)^2, \tag{7}$$

where $T_j = 1$ if class j is present in the input image, otherwise it is set to 0, m^+ is the lower limit of correct classification, i.e., the current input image belonging to class j when $\|v_j\| \in [m^+, 1]$. Similarly, m^- denotes the upper bound of error classification, which means that the current input image does not belong to class j when $\|v_j\| \in [0, m^-]$. We set m^+ and m^- to 0.9 and 0.1, respectively. Considering that the absent geographic object may stop the initial learning, we set λ to 0.5 to control its effect.

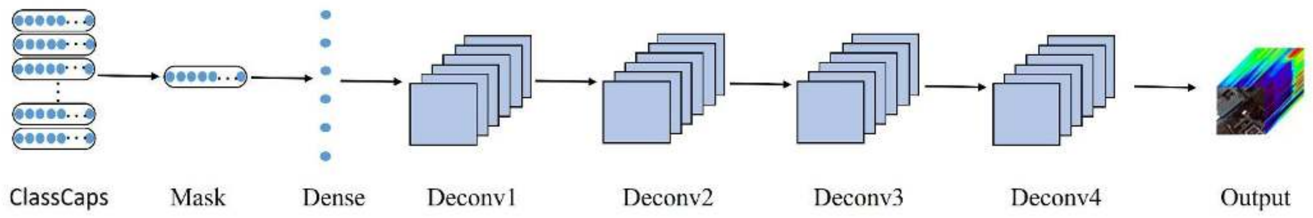


Fig. 3. Decoder network

C. Decoder network

The decoder network used in this study was composed of several deconvolutional layers (Fig. 3). It utilizes the activity vector output by the encoder network to reconstruct the input data and encourages the encoder network to extract the most relevant features of the target geographic object from the input HSI. In contrast to the original decoder network consisting of fully connected layers proposed in [40], because of the deconvolutional operations, the proposed decoder network could acquire more spatial relationships which are important for accurate HSI classification. Furthermore, only the activity vector output by the true class corresponding capsule is fed into the decoder network during the training stage in this study, which could enable the decoder network focus on the features most related to the true class. Simultaneously, the proposed decoder network had a small computational requirement that effectively reduced the training time of the model.

To effectively avoid overfitting risks, especially with a limited training sample of HSIs, a reconstruction loss was introduced as a regularization method in the training stage, and the total loss of DC-CapsNet is a sum of the margin loss and the reconstruction loss.

$$Loss_{\text{total}} = Loss_{\text{margin}} + \alpha\, Loss_{\text{recon}}, \tag{8}$$

$$Loss_{\text{recon}} = \|\mathbf{X} - \mathbf{X}_{\text{recon}}\|, \tag{9}$$

where $\mathbf{X}$ is the input data, $\mathbf{X}_{\text{recon}}$ is the reconstruction data from the decoder network, and α is a regularization factor that controls the effect of the reconstruction loss.

D. Proposed network architecture for HSI classification

The entire structure of the DC-CapsNet is a combination of the encoder and decoder networks (Fig. 4). The encoder network is the principal part of the model, which plays the role of a feature extractor, mainly responsible for extracting spectral–spatial features from input HSI data. The decoder network is a regularization method in this study which reconstructs the input HSI data to ensure that the features captured by the encoder network are the most relevant for HSI classification. Moreover, it could mitigate the overfitting issue when the number of training samples is limited in the HSI classification.

The main architecture of the DC-CapsNet for HSI classification is shown in Table I. In the encoder network, the first 2 layers adopted 3×3 convolutional kernels to extract the spectral–spatial features from the input HSI data, and the number of filters was 128 and 64, respectively. The next layer, namely, the PrimaryCaps layer, was composed of 8 convolutional capsule units, with a 3×3 kernel size, and the dimension of each convolutional capsule unit was 16. The fourth layer was a 3D convolutional capsule layer, which used 3D convolution, with a 3×3×8 kernel size to output 8 8-dimension convolutional capsule units. The last layer was a fully connected capsule layer; there were a total of *n_class* capsules, and each capsule provided a 16-dimension activity vector. The length of the activity vector represents the probability that the corresponding geographic entity exists in the current input data. Thus, the final HSI classification result can be obtained using the length of the activity vector. Furthermore, there are three routing iterations between the consecutive capsule layers in the DC-CapsNet. In the decoder network, a fully connected layer followed by four deconvolutional layers was used to reconstruct the input HSI data.

Table I Architecture of DC-CapsNet

Encoder Network				
Convolutional layer				
Layer	Kernel size	stride	Bath normalization	Activation function
L1	(3 × 3) × 128	(1, 1)	YES	Relu
L2	(3 × 3) × 64	(1, 1)	YES	Relu
Primary capsule layer				
Layer	Kernel size	stride	Bath normalization	Activation function
L3	(3 × 3) × 16 × 8	(2, 2)	YES	Relu, Squash
3D convolutional capsule layer				
Layer	Kernel size	stride	Bath normalization	Activation function
L4	(3 × 3 × 8) × 8 × 8	(2, 2, 8)	NO	Squash
Dense capsule layer				
Layer	Output size			Activation function
L5	n_{class}×16			Squash
Decoder Network				
Fully-connected layer				
Layer	Number of neurons		Bath normalization	Activation function
L6	7 × 7 × 16		YES	Relu
Deconvolutional layers				
Layer	Kernel size	stride	Bath normalization	Activation function
L7	(3 × 3) × 64	(1, 1)	NO	Relu
L8	(3 × 3) × 32	(1, 1)	NO	Relu
L9	(3 × 3) × 16	(1, 1)	NO	Relu
L10	(3 × 3) × C	(1, 1)	NO	Relu

III. EXPERIMENTAL RESULTS AND ANALYSIS

In this section, we introduced four common HSI datasets–the Kennedy Space Center (KSC), Indian Pines (IN), Pavia University (UP), and Salinas (SA) datasets to explore the performance of DC-CapsNet for HSI classification. The input data for all HSI datasets were normalized to values, with unit variance. For all datasets, the batch size was set to 32 after experiments. To speed up the convergence rate of the DC-CapsNet and avoid overfitting when the training samples were limited, the dynamic learning rate and early stopping were introduced. This means that, in the training stage, if the validation loss did not decrease after 30 epochs, the training process was stopped, and the maximum value of the training epochs was 200. If the validation loss was no longer decreasing after 10 epochs, the learning rate was decreased by half until the training process stopped or the learning rate became 0. The initial learning rate was set to 0.001 after the experiments. In addition, some regularization strategies were adopted to further improve the generalization ability of the proposed model, such as L2 norm, batch normalization (BN) [52], and reconstruction loss. Moreover, we used the He initialization method [53] as the initialization method and Relu [54] as the activation function (together) to boost the training process of the model. The overall accuracy (OA), average accuracy (AA), and kappa coefficient (K) were used to quantitatively evaluate the classification performance of the proposed model. In this paper, the experimental hardware platform was a desktop computer with an Intel i5-8500k CPU, GTX1080Ti GPU, and 64 GB memory. The software environment was composed of Windows 10 x64 as an operating system, CUDA 9.0, and cuDNN 7.4.1. Keras 2.2.4 framework using TensorFlow 1.5.0 was used as a backend, and Python 3.6.5 was the programing language.

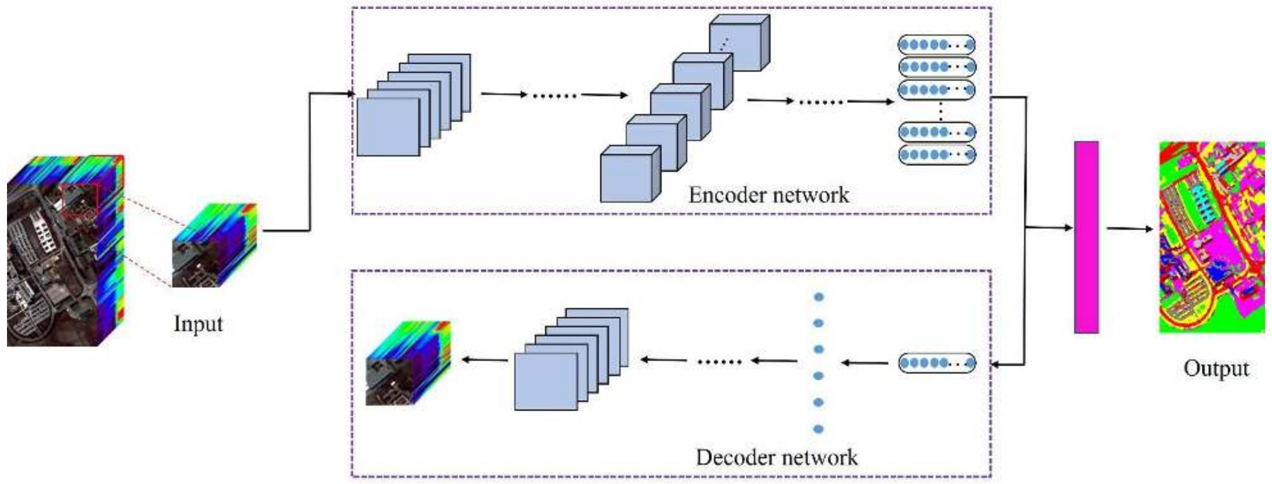


Fig. 4. Framework of DC-CapsNet for HSI classification

A. Experimental Datasets

The KSC dataset contains 176 bands. It consists of 512 × 614 pixels and 13 classes. There are a total of 200 effective bands with 145 × 145 pixels in the IN dataset, and the ground truth available was divided into 16 classes. The UP dataset contains 103 spectral bands with 610 × 340 pixels. The available land objects can be divided into nine classes. The SA dataset was consisting of 204 bands with 512 × 217 pixels. A total of 16 classes were included in the ground-truth data.

B. Influence of Parameters

In this study, the input data were the $w \times w \times L$ neighboring pixel blocks split from the original image. The larger the neighboring pixel block size, the more spatial–spectral information obtained. Thus, the neighboring pixel block size can influence the final HSI classification result, and it is an important factor for DC-CapsNet. In addition, the convolutional layers were used as feature extractors; they captured high-level features from the original input image and imported into the capsule layers. The convolutional kernel sizes used in the convolutional layers and the number of convolutional layers impact the classification performance of the model. We will explore the influence of these parameters on the performance of the DC-CapsNet. In this study, the performance of the proposed decoder network is also discussed.

It is worth mentioning that the decoder network is not introduced into the model when we evaluated the influences of the spatial size of the input cube, convolutional kernel sizes, and number of convolutional layers. In other words, only the encoder network was used in our experiments. This is because the encoder network is mainly a part of DC-CapsNet, which captures spectral–spatial features and classifies the input HSI data. In this study, we aim to select the best encoder network to extract spectral–spatial features from HSIs and then introduce the decoder network to further enhance the generalization ability of the encoder network. Furthermore, for all experiments in this study, the IN and KSC datasets randomly choose 3 % training samples, and the UP and SA datasets randomly choose 0.5 % training samples. The number of validation samples was the same as that in the training set, and the remaining samples were used to test the performance of the model.

1) Neighboring pixel block size

In this section, we discuss the classification performance for the neighboring pixel block sizes of 7, 9, 11, 13, and 15, individually. The convolutional kernel size and number of convolutional layers were set to 3 and 2. Table II shows the classification results for the four datasets. It can be seen that the OA increases as the neighboring pixel block size gradually increases, and a threshold effect exists. For KSC, IN, and UP datasets, the changing trend of classification accuracy is similar. When the neighboring pixel block size was 9 and 13, the accuracy will increase greatly. The highest OA was obtained when the neighboring pixel block size was set to 13. However, the classification accuracy decreases when we chose a 15 × 15 spatial size of the pixel block. For SA, the performance of classification was optimal when the neighboring pixel block size was 15, but the improvement was very small when the neighboring pixel block size was 13. Therefore, we chose 13 as the neighboring pixel block size for the DC-CapsNet. Fig. 5 and Fig. 7 show the tendencies in loss and accuracy of the training and validation sets for different neighboring pixel block sizes over the IN and UP datasets. Similarly, with the increase in the spatial size of the input cube, the changes in loss and accuracy were more stable, and they were the most stable when the neighboring pixel block size was 13.

Table II Overall accuracy (%) for different spatial size of neighboring pixel blocks

Data Set	7	9	11	13	15
KSC	81.30	88.86	91.33	94.86	93.37
IN	81.83	87.71	87.90	93.94	90.75
UP	90.32	93.04	93.65	96.46	94.28
SA	91.42	93.02	93.81	95.76	95.78

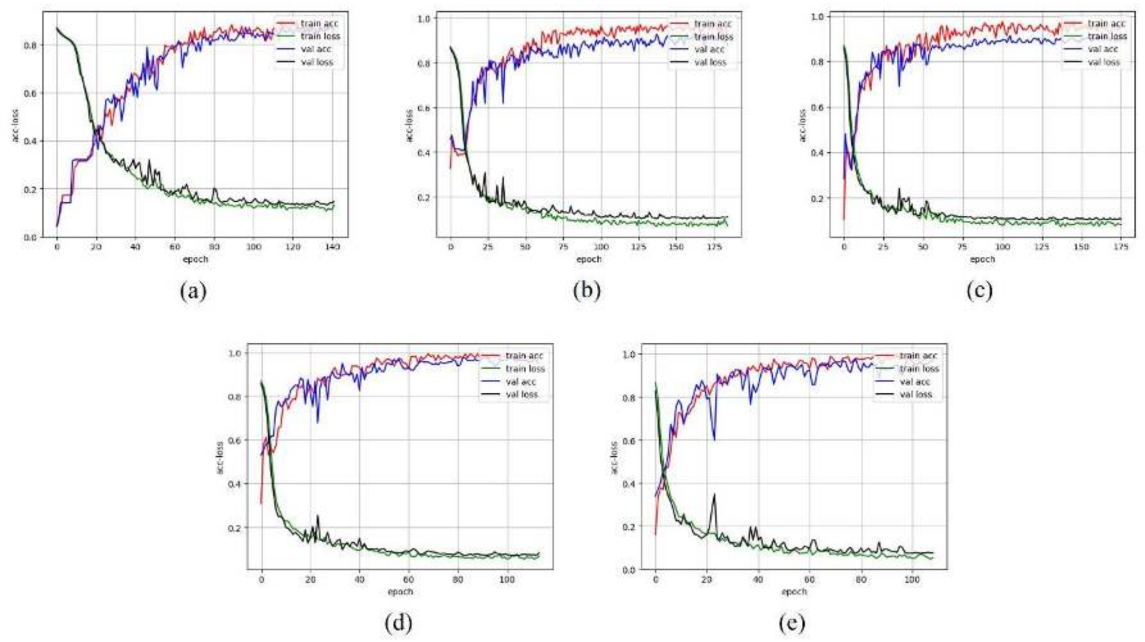


Fig. 5. Tendencies in loss and accuracy of training and validation sets under different neighboring pixel block size over Indian Pines dataset for different neighboring pixel block sizes: (a) 7, (b) 9, (c) 11, (d) 13 and (e) 15.

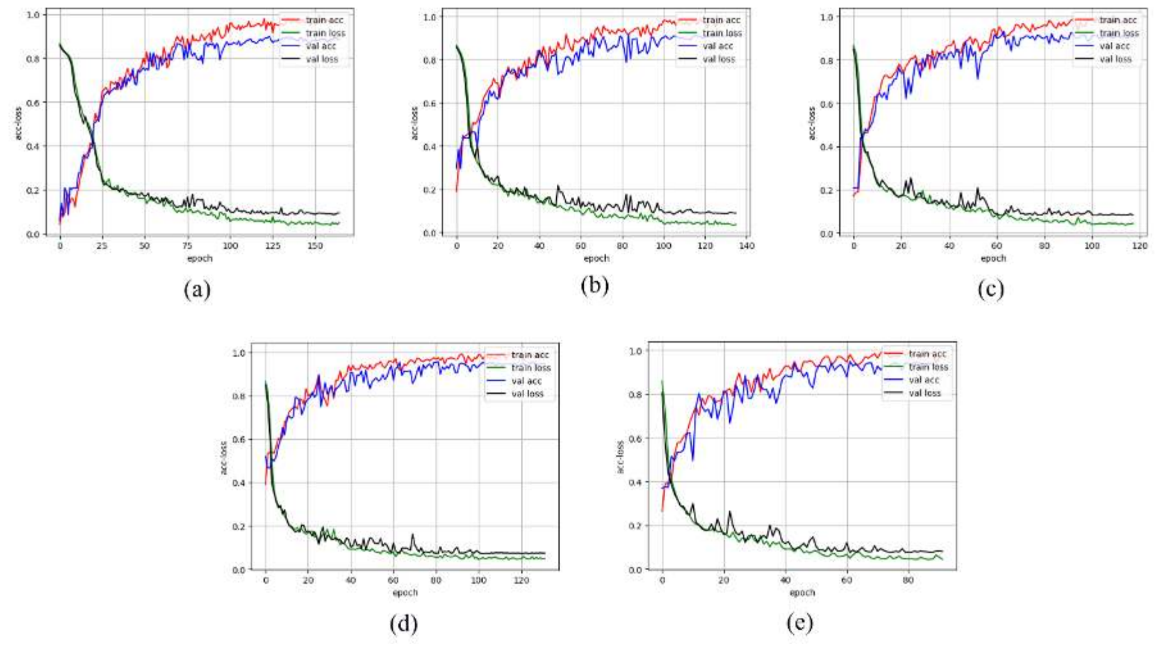


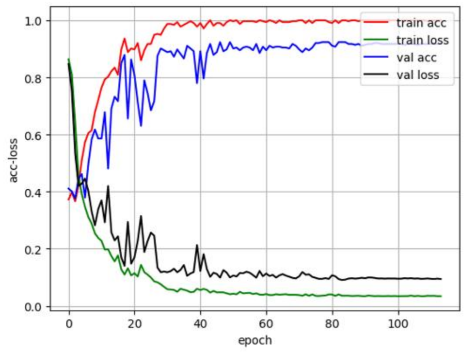
Fig. 6. Tendencies in loss and accuracy of training and validation sets under different neighboring pixel block size over Pavia University dataset for different neighboring pixel block sizes: (a) 7, (b) 9, (c) 11, (d) 13 and (e) 15.

2) *Convolutional kernel sizes*

In this section, the influences of different convolutional kernel sizes are explored. Owing to the limitation of the input cube sizes, we only discuss the classification result for the convolutional kernel sizes of 3 and 5. The neighboring pixel block size and the number of convolutional layers were set to 13 and 2, respectively. The classification accuracy is shown in Table III. When the convolution kernel size is 3, the OAs of the model on KSC, IN, UP, and SA datasets reached 94.86%, 93.94%, 96.46%, and 95.76% respectively, which is 5.51%, 5.77%, 0.38%, and 1.86% higher than that of the convolution kernel of 5. Fig. 7 and Fig. 8 show the tendencies in the loss and accuracy of the training and validation sets for different convolutional kernel sizes over the IN and UP datasets. It is obvious that the classification result of the DC-CapsNet is optimal when the convolutional kernel size is 3. Moreover, it makes the optimization procedure more stable and the convergence speed faster.

Table III Overall accuracy (%) for different convolutional kernel sizes and different number of convolutional layers

Data Set	Kernel size		No. convolutional layers		
	3	5	1	2	3
KSC	94.86	89.35	94.48	94.86	90.46
IN	93.94	88.17	92.24	93.94	89.36
UP	96.46	96.08	95.28	96.46	93.41
SA	95.76	93.90	95.61	95.76	94.37


(a)

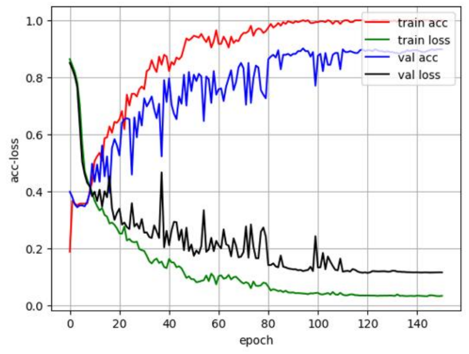
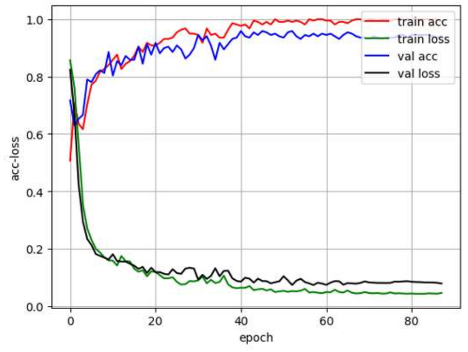

(b)

Fig. 7. Tendencies in loss and accuracy of training and validation sets under different convolutional kernel sizes over Indian Pines dataset for neighboring pixel block sizes (a) 3 and (b) 5.


(a)

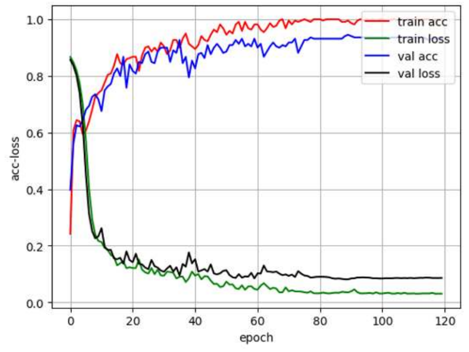
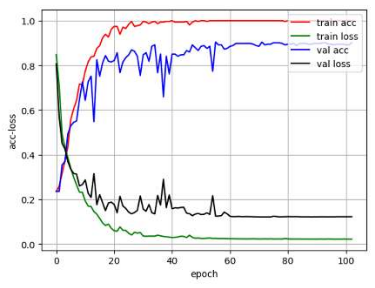
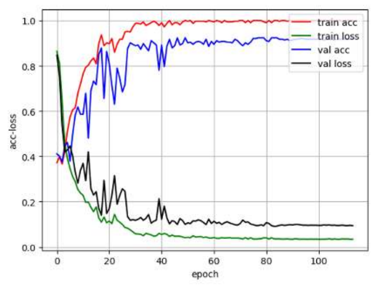

(b)

Fig. 8. Tendencies in loss and accuracy of training and validation sets under different convolutional kernel sizes over Pavia University dataset for neighboring pixel block sizes (a) 3 and (b) 5.


(a)


(b)

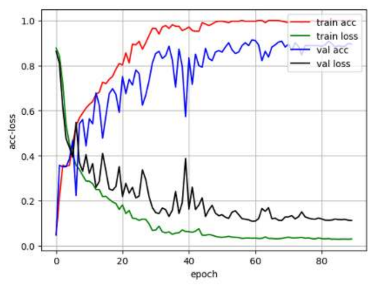

(c)

Fig. 9. Tendencies in loss and accuracy of training and validation sets under different number of convolutional layers over Indian Pines dataset for neighboring pixel block sizes of (a) 1, (b) 2 and (c) 3.

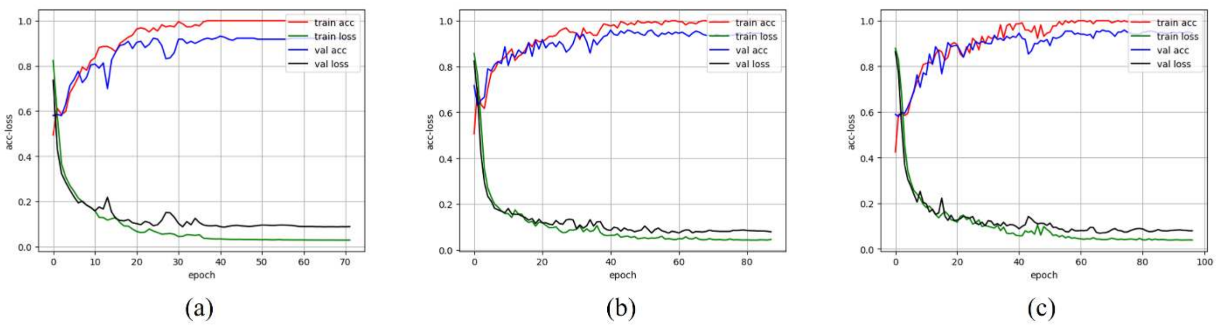


Fig. 10. Tendencies in loss and accuracy of training and validation sets under different number of convolutional layers over Pavia University dataset for neighboring pixel block sizes of (a) 1, (b) 2 and (c) 3.

3) *Number of convolutional layers*

We analyzed the performance of the model in four datasets when the number of convolutional layers were 1, 2, and 3. The neighboring pixel block size and convolutional kernel sizes were set to 13 and 3, respectively. As shown in Table III, the best classification result was provided by using two convolutional layers, that is 94.86 %, 93.94 %, 96.46 %, and 95.76 % respectively on the KSC, IN, UP and SA datasets. As the number of convolutional layers increased to 3, the accuracy of the DC-CapsNet decreased significantly, by 4.8 %, 4.58 %, 3.5 %, and 1.39 % on the four datasets. Fig. 9 and Fig. 10 show the tendencies in the loss and accuracy of the training and validation sets for different convolutional kernel sizes over the IN and UP datasets. We found that one convolutional layer could not capture the robust spectral–spatial features effectively, and three convolutional layers led to the worst result. Therefore, we deduce that using two convolutional layers in shallow layers to extract features from the original HSI data may be a good choice.

4) *Selection of decoder network*

In this study, a decoder network was used to reconstruct the input image using the final output activity vector. The selection of the decoder network in the model is very important. Thus, we consider the parameters numbers of the original decoder network represented in [40] and the proposed decoder network in this study over KSC, IN, UP, and SA datasets, as well as the classification results when adopting the original decoder network, proposed decoder network, and no decoder network under the condition of different ratios of the training set. A 13 × 13 spatial size of the input cube, kernel size of 3, and 2 convolutional layers are used in this section. The weight coefficient of reconstruction loss α is set to 0.0005. Table IV shows the parameters numbers of different kind of decoder network. Obviously, the original decoder network has a large number of parameters because of the fully connected layers. Due to the difference of the number of spectral dimensions, the parameters of the original decoder network are quite different over KSC, IN, UP, and SA datasets. The original decoder network has the least number of parameters in the UP dataset, which is more than 18 million, and has the most parameters in the SA dataset—nearly 36 million. In contrast, the proposed decoder network consists of deconvolutional layers, which is much lighter than the original decoder network. On the four HSI datasets, the parameters of the proposed decoder network are only 75340 at most, less than 1 / 400th of the original decoder network.

The classification results are listed in Table V. The original decoder network results in overfitting when the number of training samples is limited owing to a large number of trainable parameters. It has limited improvement in classification accuracy and even makes the model performance of classification worse when the number of training samples is insufficient. In particular, for the IN dataset (having uneven sample distribution), the OA of the model can be significantly reduced by using the original decoder network when the ratios of the training set are 0.5 %, 1 %, and 3 %. Similarly, because we chose 0.5 % training sample on the UP and SA datasets, the original decoder network will also lead to a decrease in the OA of the model. The proposed decoder network, by contrast, can significantly improve the classification accuracy. The improvement in the classification performance was particularly obvious when the training samples were relatively small. When the ratios of training samples are set to 0.5 %, the DC-CapsNet with the proposed decoder network has highest OA that reached values of 77.54 %, 72.76 %, 96.71 %, and 97.14 % on KSC, IN, UP, and SA datasets, receptively. With the increase of the training samples, the decoder network's improvement in classification accuracy decreased slightly. When the proportion of the training set was greater than 5 % for UP and SA datasets and 7 % for KSC and IN datasets, the advantages of the proposed decoder network became less obvious. As the ratios of the training set are 10 % for KSC, IN, UP, and SA datasets, the accuracy of DC-CapsNet with the proposed decoder network receptively reached 99.57 %, 99.16 %,99.89 %, and 99.98 %, with an improvement of 0.21 %, 0.05 %, 0.02 %, and 0.03 %. This is because the performance of deep learning models is usually related to the number of training samples. The classification accuracy increases rapidly with the increasing training samples, even higher than 99% so that the improvement of the proposed decoder network is not obvious.

Table IV The parameters of different kind of decoder network

Data Set	original	proposed
KSC	31,119,920	71,280
IN	35,301,896	74,760
UP	18,441,727	60,695
SA	35,994,796	75,340

Table V overall accuracy (%) for different kind of decoder network

Data Set	Decoder network	0.5 %	1 %	3 %	5 %	7 %	10 %
KSC	no	73.61±2.92	90.07±2.37	94.86±1.30	96.26±1.77	98.49±0.45	99.36±0.18
	original	74.04±11.99	90.44±1.52	95.51±0.52	97.63±0.32	98.79±0.57	99.42±0.15
	proposed	77.54±0.62	91.47±0.44	95.97±1.16	97.73±0.49	98.90±0.44	99.57±0.07
IN	no	72.40±1.62	79.32±2.27	93.94±1.30	96.06±0.82	98.11±0.10	99.11±0.28
	original	71.12±1.61	79.09±1.75	93.66±0.94	96.61±0.62	98.14±0.25	99.13±0.16
	proposed	72.76±1.61	79.65±1.64	94.42±1.11	96.43±0.65	98.15±0.11	99.16±0.23
UP	no	96.46±0.27	97.76±0.14	99.26±0.08	99.69±0.06	99.79±0.04	99.87±0.03
	original	95.82±0.68	97.86±0.14	99.33±0.06	99.71±0.02	99.81±0.03	99.87±0.04
	proposed	96.71±0.38	97.92±0.25	99.48±0.12	99.76±0.06	99.84±0.04	99.89±0.04
SA	no	95.76±0.70	96.50±0.46	98.91±0.13	99.81±0.03	99.88±0.07	99.95±0.03
	original	95.24±1.11	96.95±0.16	98.93±0.12	99.82±0.07	99.90±0.06	99.97±0.00
	proposed	97.14±0.32	97.79±0.31	99.27±0.09	99.85±0.04	99.91±0.05	99.98±0.00

To better understand the effect of the proposed decoder network, we further investigated the tendencies in loss and accuracy of the training and validation sets for the model using a different type of decoder network on the IN and UP datasets, with a specific number of training samples (for the IN dataset, we chose 3 % training samples; for the UP dataset, we chose 0.5 % training samples). As shown in Fig. 11 and Fig. 12, the introduction of the decoder network improved the robustness of the model. Furthermore, the performance of the proposed decoder network outperformed the original network. It is noteworthy that the loss and accuracy curves of the model introducing the proposed decoder network fluctuated relatively greatly on the IN dataset, but the final result was optimal and effective.

McNemar's test [55] is a statistical procedure that can be used to compare two classification models that are dependent or correlated. To better show the effectiveness of the proposed decoder network, we performed some statistical analysis, using McNemar's test, on the SA dataset (chose 3 % training samples and 10 % training samples) for the no decoder and the original decoder network, as well as the original decoder and the proposed decoder network, respectively. The contingency tables of these three cases are listed in Table VI-IX. By calculation, the corresponding probability p is 1.57×10^{-16} and 1.92×10^{-19} respectively when we chose 10 % training samples. At the significance level of 0.05, the data provide sufficient evidence to conclude that the performance of the model is significantly different when using different kinds of decoder networks. In other words, the decoder network improves the classification accuracy of the model, to a certain extent, and the effect of the proposed decoder network in this paper is better than that of the original decoder network. When the ratio of the training set is 10 % for SA dataset, the p is 1.94×10^{-8} and 0.27 respectively. It means that the decoder network can improve the final classification result using the 0.05 significance level, but the performance of the original decoder network and the proposed decoder network is almost the same. It is because the deep learning models are data-driven approaches. When the number of training samples is sufficient, we can get a better original decoder during the training stage. On the other hand, the classification accuracy is higher than 99.95 %, it is hard to make a big improvement. Thus, the performance difference between the proposed decoder network and the original decoder network is not obvious when the number of training samples is relatively large.

Table VI The contingency table for the no decoder and the proposed decoder network over SA dataset with 3% training samples

Original decoder	No decoder		Total
	Correct	Incorrect	
Correct	50148	217	50365
Incorrect	75	421	496
Total	450223	638	50861

Table VII The contingency table for the original decoder and the proposed decoder over SA dataset with 3% training samples

Proposed decoder	Original decoder		Total
	Correct	Incorrect	
Correct	50291	233	50524
Incorrect	74	263	337
Total	50365	496	50861

Table VIII The contingency table for the no decoder and the original decoder over SA dataset with 10% training samples

Original decoder	No decoder		Total
	Correct	Incorrect	
Correct	43249	34	43283
Incorrect	2	8	10
Total	43251	42	43293

Table IX The contingency table for the proposed decoder and the original decoder over SA dataset with 10% training samples

Proposed decoder	Original decoder		Total
	Correct	Incorrect	
Correct	43274	9	43283
Incorrect	4	6	10
Total	43278	15	43293

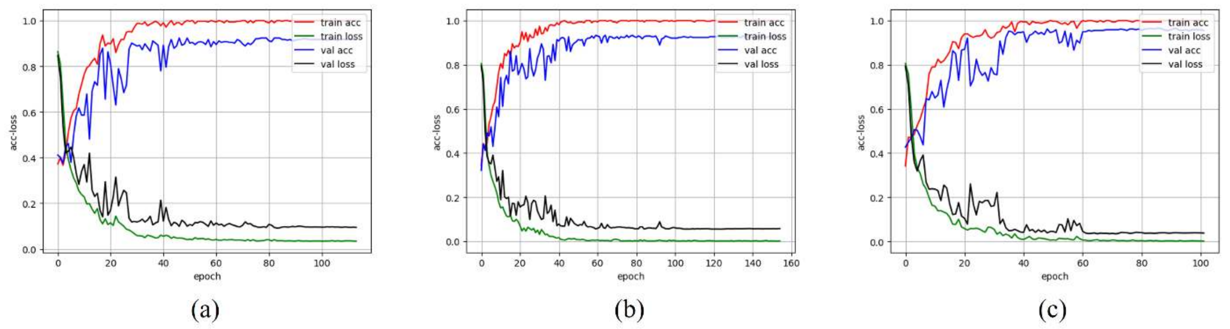


Fig. 11. Tendencies in loss and accuracy of training and validation sets under different kind of decoder network over Indian Pines dataset: (a) no decoder, (b) original decoder and (c) proposed decoder

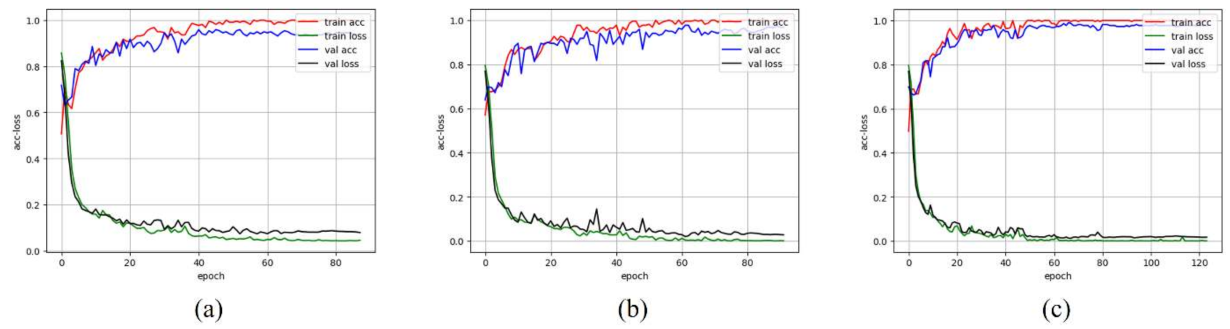


Fig. 12. Tendencies in loss and accuracy of training and validation sets under different kind of decoder network over Pavia University dataset: (a) no decoder, (b) original decoder and (c) proposed decoder

5) Weight coefficient of reconstruction loss

In this paper, the reconstruction loss output by the decoder network is a regularization method during the training stage, and its weight coefficient α directly influent the performance of the model. In this part, we observed the influences of the reconstruction loss on the classification performance of the proposed model when α was 0.0001, 0.0003, 0.0005, 0.0007, 0.001, 0.003, 0.005, 0.007, 0.01, 0.03, 0.05, 0.1, 0.3, 0.5, 0.7, and 1. The neighboring pixel block size, convolutional kernel sizes, and convolutional layers were set to 13, 3, and 2, respectively. The classification results as shown in Fig. 13. The results on the KSC and UP datasets first improve significantly as α increases and then degrade slightly. DC-CapsNet provides the highest OA when the weight coefficient was 0.0005, namely 95.97 % and 96.71 % receptively. For the IN and SA datasets, we observe that the OA declines at first and then increase significantly. Similar to the other two HSI datasets, the performance of classification was optimal when α was 0.0005. Moreover, when α continued to improve on all four datasets, OA declines dramatically and then fluctuated lightly. This is because the encoder network is mainly a part of DC-CapsNet, which is responsible for extracting spectral–spatial features from input HSI data. The decoder network is a regularization method which the class capsule to encode the spectral-spatial information of the target geographic object. Appropriate regularization term can enhance the generalization ability of the model. When the regularization term is too large, the performance of the model will be adversely affected. By contrast, when the regularization term is too small, it can't give full play to its role, and cannot effectively enhance the performance of the model. Thus 0.0005 may be a good choice for the weight coefficient of reconstruction loss. The reconstruction loss does not dominate the margin loss during training stage on the one hand, on the other hand, it could improve the classification accuracy of DC-CapsNet.

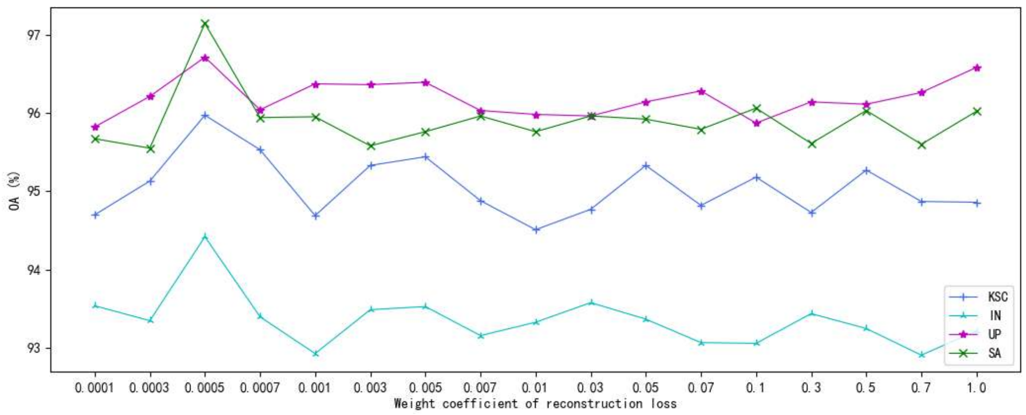


Fig. 13. Overall accuracy (%) under different weight coefficients over KSC, IN, UP, and SA dataset

C. Experiment Results and Analysis

In our study, the KSC, IN, UP, and SA datasets were used to evaluate the performance of the DC-CapsNet. We chose the model having 2 traditional convolutional layers in shallow layers, and the convolutional kernel sizes were set to 3. The weight coefficient of the reconstruction loss generated by the decoder network was set to 0.0005. The spatial size of the input HSI cube was selected as 13 × 13. To further explore the potential of DC-CapsNet in the HSI classification field, we compared it with SVM [12], 3D-CNN [29], SSRN [30], deep feature dense network (DFDN) [32], and 3D-CapsNet [43] models on the KSC, IN, UP, and SA datasets. For the KSC and IN datasets, the ratios of training samples and validation samples were set to 3 %. For the UP and SA datasets, the ratios of training samples and validation samples were set to 0.5 %. For 3D-CapsNet, we adjusted the number of feature maps output by the PrimaryCaps layer under the condition of existing training samples and selected the parameters with the best performance. With the exception of 3D-CapsNet, the architecture of these models of SVM [12], 3D-CNN [29], SSRN [30] and DFDN [32] was the same as in their papers.

Table X shows the classification results of all models on the KSC, IN, UP, and SA datasets. The classification of DC-CapsNet has the highest classification accuracy on the KSC dataset. The highest OA, AA, and K reached values of 95.97 %, 93.43 %, and 0.9551, with an improvement of 2.69 %, 1.81 % and 0.03 for the SSRN, respectively. For the IN dataset, the proposed method performed better than other compared models, which improved the OA, AA and K of the SSRN by 2.93 %, 4.8 % and 0.034, respectively. For the UP dataset, our proposed model achieved the best classification result; compared with the SSRN, the OA, AA and K increased by 1.48 %, 1.77 % and 0.0188, respectively. A similar result was observed for the SA dataset. The OA, AA and K values of our model were found to be 97.14 %, 98.06 % and 0.9682, respectively, which are higher than those of other state-of-the-art methods. Furthermore, the SSRN obtains the second-best result among all the datasets, while the SVM performance was particularly poor. The SSRN involves several spectral residual blocks and spatial residual blocks, which provide a lighter and deeper network structure to classify HSI data. Thus, the SSRN can achieve a better classification result than 3D-CapsNet, 3D-CNN and DFDN. Moreover, both 3D-CapsNet and 3D-CNN have a relatively shallow network architecture. We found that the performance of 3D-CapsNet is generally better than that of 3D-CNN by comparing their classification results. However, the 3D-CNN is superior to 3D-CapsNet on the KSC dataset. This is because there are too many parameters of 3D-CapsNet that easily lead to overfitting, while the training samples of the KSC dataset are less than those of the other datasets. Similar results also arise with the DFDN.

Table X Classification results of different models on Kennedy Space Center, Indian Pines, Pavia University and Salinas

Data set		**SVM**	**3D-CNN**	**SSRN**	**DFDN**	**3D-CapsNet**	**DC-CapsNet**
KSC	OA (%)	81.83±0.04	87.65±1.89	93.28±1.25	88.43±0.88	83.63±1.26	95.97±1.16
	AA (%)	73.86±2.33	85.69±2.40	91.62±1.02	87.58±1.44	78.80±1.20	93.43±1.84
	K×100	79.73±0.05	86.24±2.11	92.51±1.41	87.11±0.96	81.74±1.42	95.51±1.29
IN	OA (%)	53.55±1.70	79.15±1.20	91.49±1.14	81.89±3.10	80.44±2.40	94.42±1.11
	AA (%)	31.26±2.36	79.79±6.54	86.93±6.51	81.07±1.91	79.66±1.07	91.73±1.11
	K×100	43.23±2.45	76.17±1.38	90.24±1.31	79.61±3.41	77.57±2.78	93.64±1.26
UP	OA (%)	78.53±0.74	86.55±0.97	95.23±0.57	88.77±1.47	91.73±0.46	96.71±0.38
	AA (%)	69.94±0.91	82.76±2.07	93.74±0.52	86.06±1.36	88.76±0.89	95.51±0.41
	K×100	70.68±0.92	81.96±1.29	93.75±0.77	84.96±2.01	88.96±0.62	95.63±0.50
SA	OA (%)	83.69±1.39	87.81±1.72	95.29±0.26	88.80±1.78	89.69±2.29	97.14±0.32
	AA (%)	86.34±2.05	92.18±1.81	97.40±0.13	90.53±2.24	94.19±0.71	98.06±0.43
	K×100	81.75±1.57	86.36±1.96	94.76±0.28	87.51±1.99	88.53±2.52	96.82±0.35

To further consider the performance of the DC-CapsNet, the tendencies in loss and accuracy of the training and validation sets for comparison of deep learning models and DC-CapsNet on the four datasets were observed (Fig. 14–17). For DFDN, it was obvious that there was a large fluctuation in the loss on the validation set, and overfitting occurred on all four datasets. Moreover, compared with 3D-CNN, SSRN, DFDN and 3D-CapsNet, DC-CapsNet had better convergence results and achieved the most stable loss and accuracy curves on the validation set.

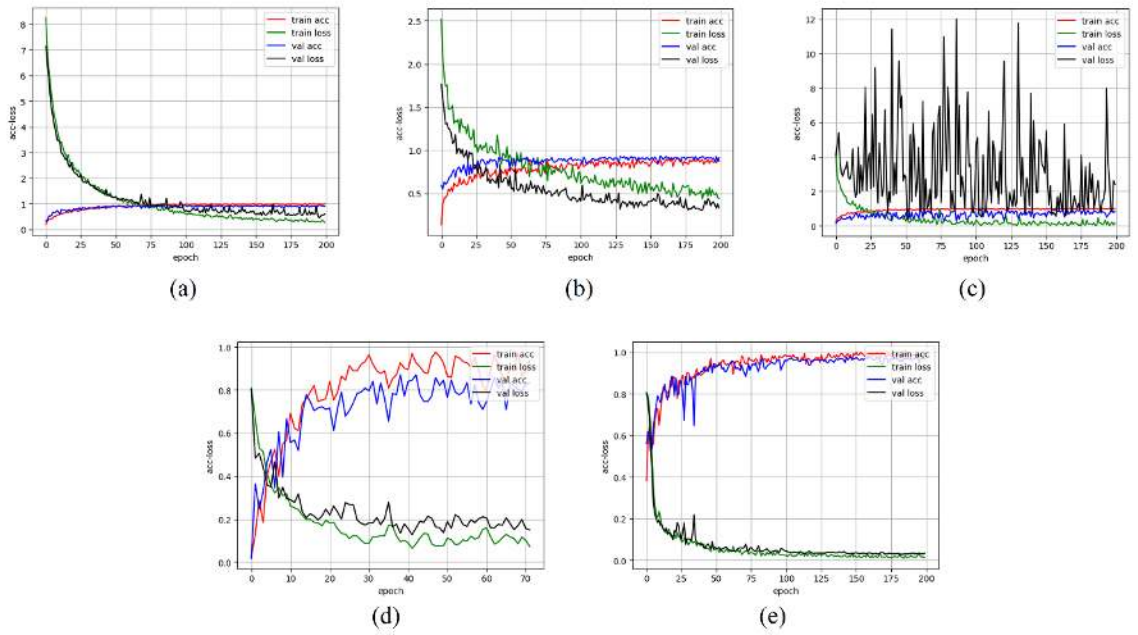


Fig. 14. Tendencies in loss and accuracy of training and validation sets under different models over Kennedy Space Center dataset: (a)–(e) classification results of 3D-CNN, SSRN, DFDN, 3D-CapsNet and DC-CapsNet

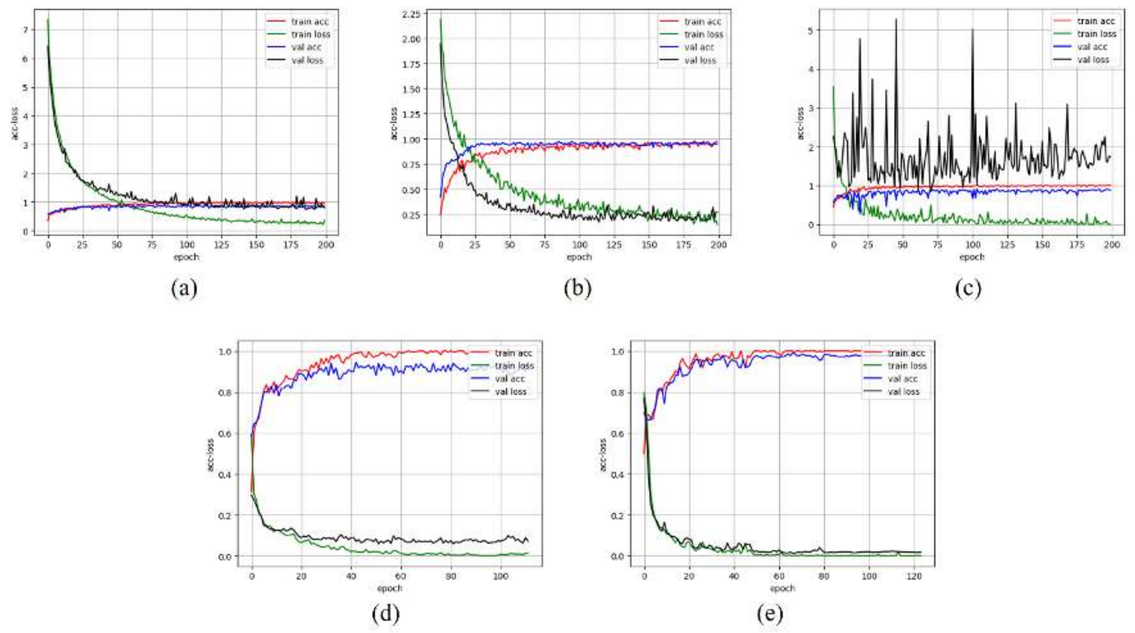


Fig. 15. Tendencies in loss and accuracy of training and validation sets under different models over Indian Pines dataset: (a)–(e) classification results of 3D-CNN, SSRN, DFDN, 3D-CapsNet and DC-CapsNet

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3101511, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

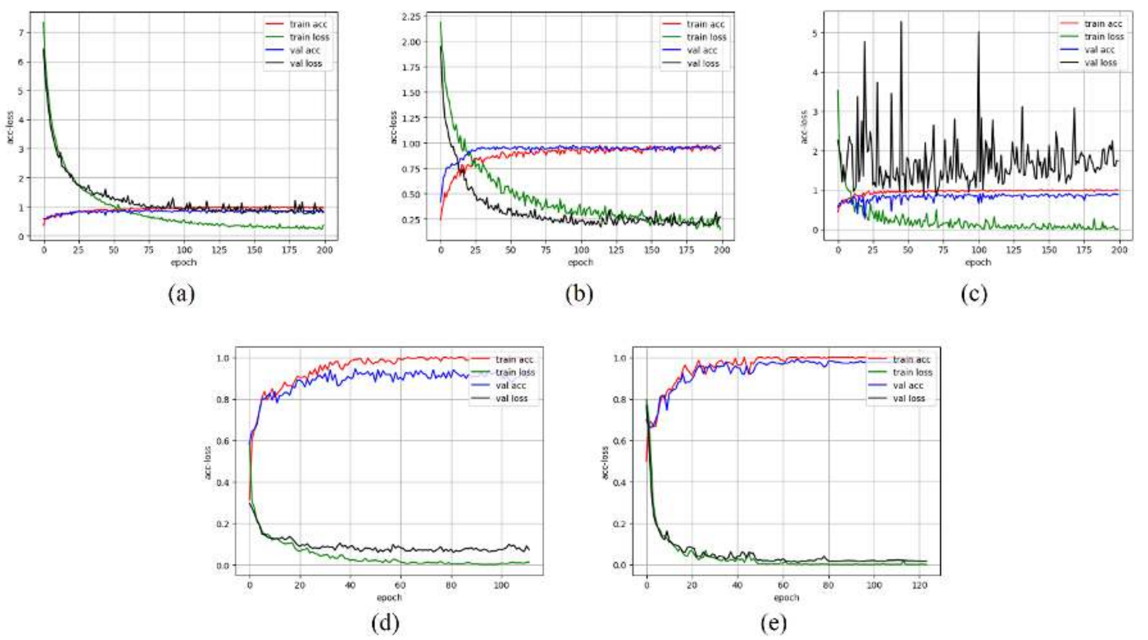


Fig. 16. Tendencies in loss and accuracy of training and validation sets under different models over Pavia University dataset: (a)–(e) classification results of 3D-CNN, SSRN, DFDN, 3D-CapsNet and DC-CapsNet

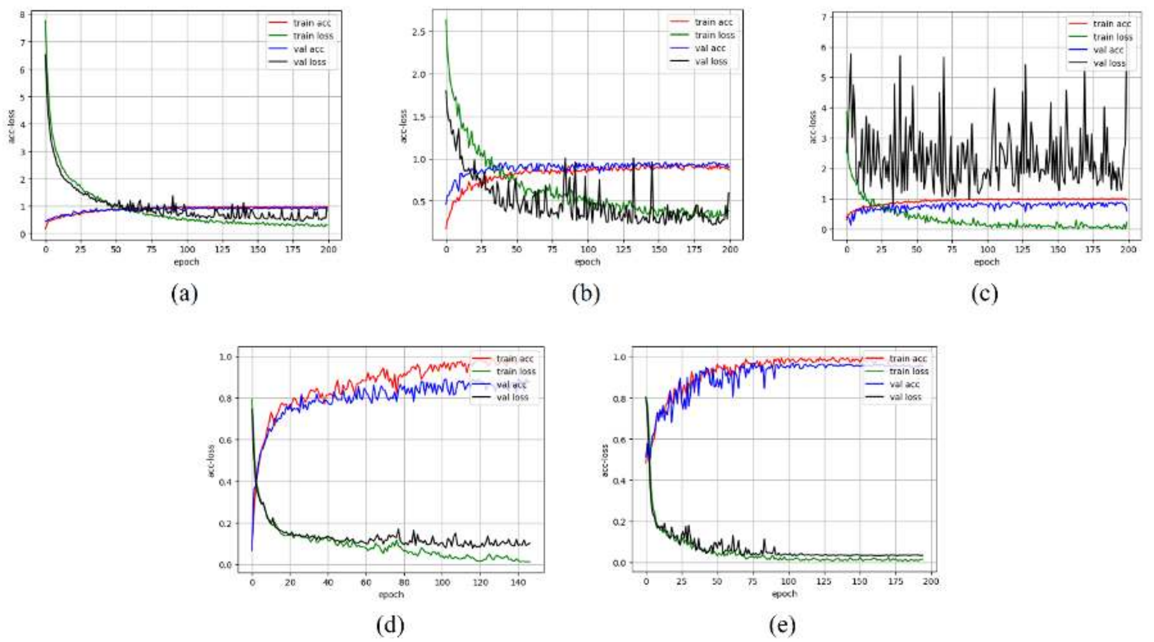


Fig. 17. Tendencies in loss and accuracy of training and validation sets under different models over Salinas dataset: (a)–(e) classification results of 3D-CNN, SSRN, DFDN, 3D-CapsNet and DC-CapsNet

Fig. 18–21 show the classification maps of the comparison models and DC-CapsNet on the KSC, IN, UP and SA datasets. We found that the classification results of the SVM contained more noise than the other methods. It is a common problem with spectral-based methods because only spectral information is considered in the HSI classification. Generally, models based on spectral–spatial information, both CNN-based and CapsNet-based methods, can generate better results than spectral-based approaches. The SSRN results in the classification maps were better than those obtained by using 3D-CapsNet and other CNN-based models. Nevertheless, the classification map obtained using DC-CapsNet is more accurate and smoother than other methods.

The training and test times indicate the computational efficiency of the model. As reported in Table XI, the computation cost of SVM is far less than that of deep learning models. The DFDN takes more time for training than other deep learning models because of the complex network structure; moreover, its test time is also much longer than that of other models. Compared with SSRN and 3D-CapsNet, DC-CapsNet converges fast on the IN and UP datasets, but not on the KSC and SA datasets. It may be attributed to the fact that some classes in these images have similar spectral-spatial property, which may cause the model to need more training time to learn the differences between these classes and provide a better classification result. Furthermore, DC-CapsNet requires more training and test times than 3D-CNN.

We could also observe the number of parameters of the different deep learning networks from Table XI. For CapsNet-based models, namely 3D-CapsNet and DC-CapsNet, it is obvious that the DC-CapsNet is much lighter than 3D-CapsNet due to the 3D convolutional capsule layer. Especially, the parameter quantity of DC-CapsNet is two orders of magnitudes lower than that of 3D-CapsNet on the KSC, IN, and SA datasets. Compared with CNN-based models, DC-CapsNet works well with the fewer parameters than 3D-CNN and DFDN. However, the parameters of DC-CapsNet are about 100 thousand more than that of SSRN. In the following work, we will try to reduce the number of parameters and the computation cost of DC-CapsNet under the premise of ensuring classification accuracy.

Table XI Training, test time and parameters under different models

Data Set	**Methods**	**SVM**	**3D-CNN**	**SSRN**	**DFDN**	**3D-CapsNet**	**DC-CapsNet**
KSC	Train (s)	0.01	36.41	60.08	798.56	72.45	98.40
	Test (s)	0.11	1.54	5.29	35.49	6.93	2.96
	Parameters	-	2,087,553	309,845	1,244,410	23,366,784	409,728
IN	Train (s)	0.06	64.73	112.01	1740.35	239.37	96.21
	Test (s)	0.13	3.13	3.88	68.84	14.06	6.13
	Parameters	-	2,401,756	346,784	1,247,776	30,745,728	449,664
UP	Train (s)	0.03	46.24	67.65	659.64	101.68	47.42
	Test (s)	0.83	8.67	11.39	151.76	41.89	20.01
	Parameters	-	832,349	199,153	1,239,922	1,436,800	309,248
SA	Train (s)	0.05	61.44	125.94	1562.30	250.48	129.61
	Test (s)	2.54	18.86	26.63	373.50	79.32	32.65
	Parameters	-	2,401,756	352,928	1,247,776	31,925,376	454,272

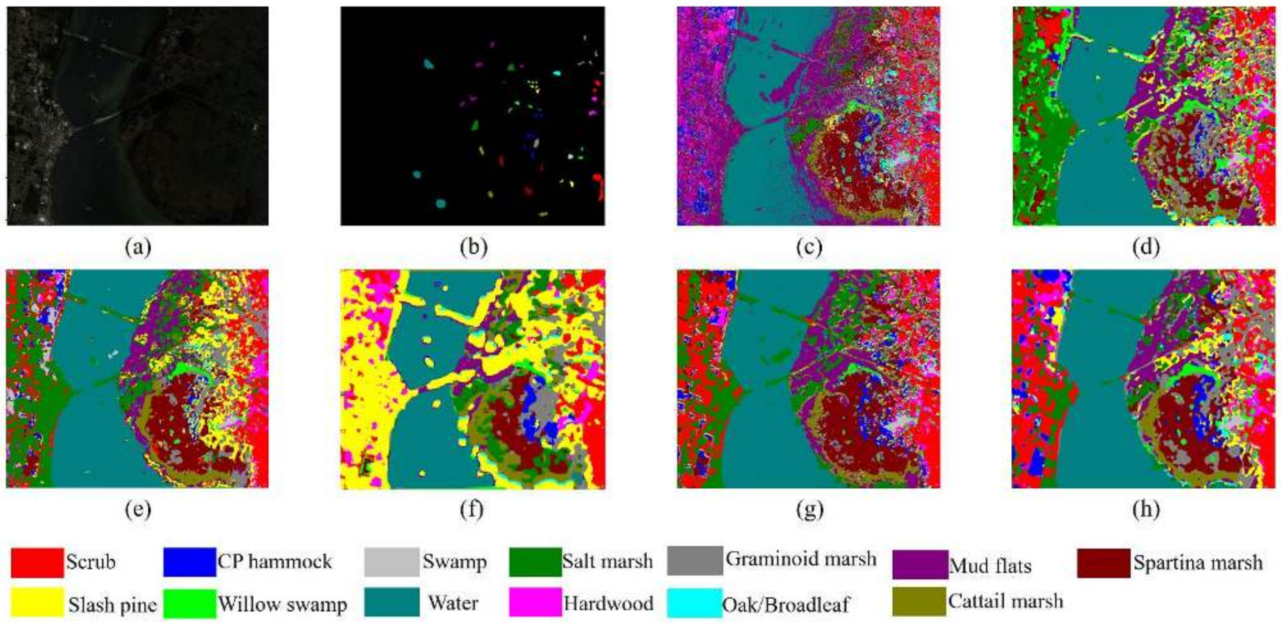


Fig. 18. Classification results of different models for Kennedy Space Center dataset: (a) false color image, (b) ground-truth labels, (c)–(h) classification results of SVM, 3D-CNN, SSRN, DFDN, 3D-CapsNet and DC-CapsNet

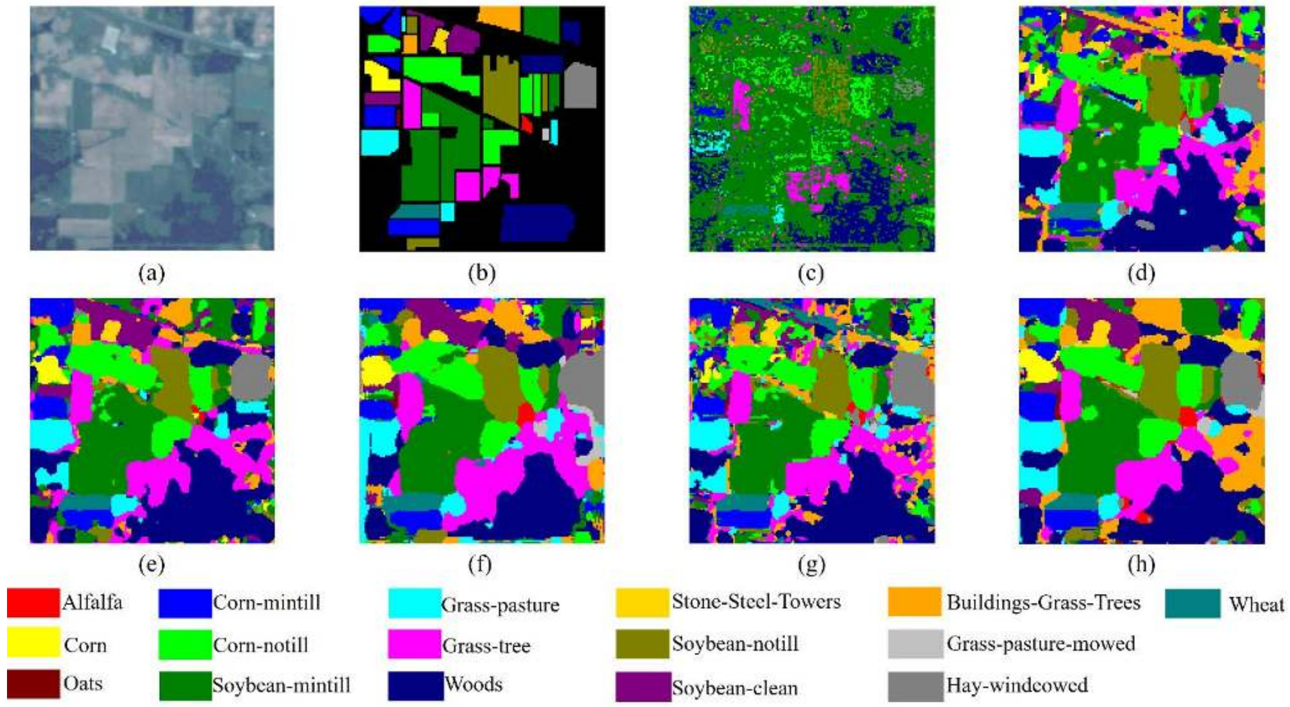


Fig. 19. Classification results of different models for Indian Pines dataset: (a) false color image, (b) ground-truth labels, (c)–(h) classification results of SVM, 3D-CNN, SSRN, DFDN, 3D-CapsNet and DC-CapsNet

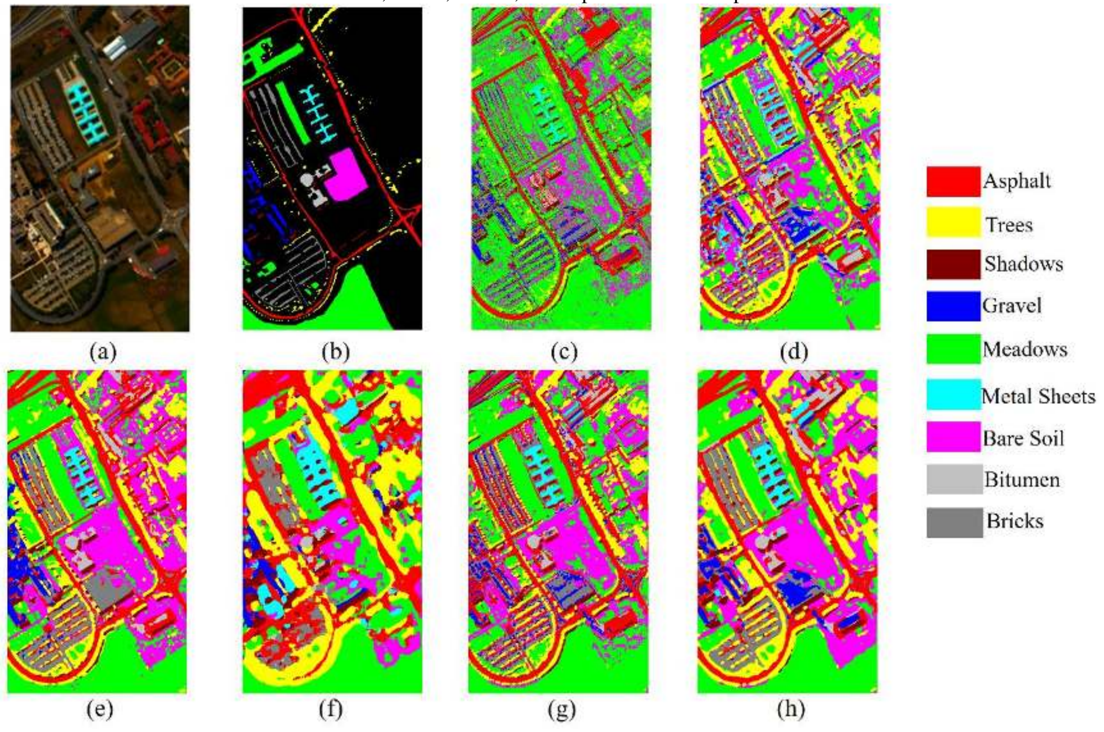


Fig. 20. Classification results of different models for Pavia University dataset: (a) false color image, (b) ground-truth labels, (c)–(h) classification results of SVM, 3D-CNN, SSRN, DFDN, 3D-CapsNet and DC-CapsNet

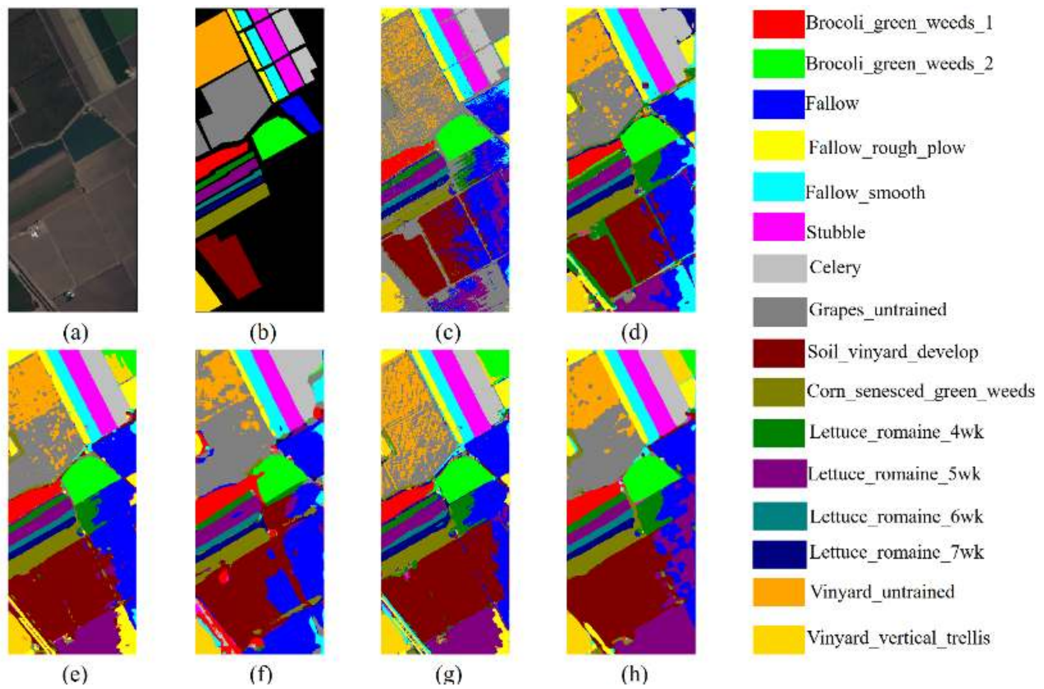


Fig. 21. Classification results of different models for Salinas dataset: (a) false color image, (b) ground-truth labels, (c)–(h) classification results of SVM, 3D-CNN, SSRN, DFDN, 3D-CapsNet and DC-CapsNet

To evaluate the generalization ability and robustness of the proposed method, we randomly selected 0.5 %, 1 %, 3 %, 5 %, 7 % and 10 % labeled samples as the training set for the KSC, IN, UP and SA datasets. Fig. 22 illustrates the OAs of the compared models and DC-CapsNet. Compared with the SVM, the classification results generated using deep learning models were superior. Moreover, the OA of deep learning models increased rapidly as the number of training samples increased. DC-CapsNet provides better OA than other deep learning models, especially under the condition of a small training set. Nevertheless, when the proportion of the training set was greater than 7 %, the advantages of DC-CapsNet over other deep learning models became less obvious. This is because the performance of deep learning models is usually related to the number of training samples. 3D-CNN, SSRN, DFDN and 3D-CapsNet have achieved relatively high classification accuracy because of the increase in the number of training samples. Therefore, the accuracy improvement of the DC-CapsNet was limited.

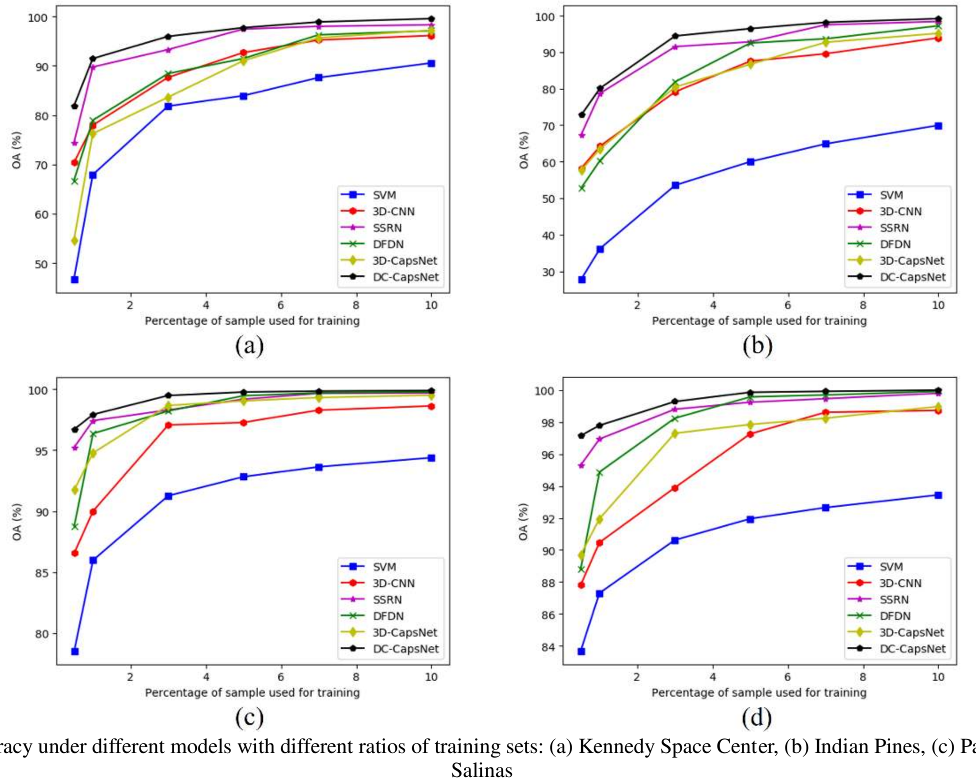


Figure 22. Overall accuracy under different models with different ratios of training sets: (a) Kennedy Space Center, (b) Indian Pines, (c) Pavia University and (d) Salinas

IV. Conclusion

In this paper, we proposed a novel deep convolutional capsule network named DC-CapsNet for the HSI classification. This model can effectively extract deep spectral–spatial features from HSIs in an end-to-end manner. Dynamic routing based on 3D convolution, which is the core of the convolutional capsule layer, is introduced to reduce the trainable parameters and enhance the robustness of the CapsNet. It uses a set of lower-level capsules rather than a single capsule to predict a higher-level capsule, which allows the model to capture more discriminative spectral–spatial features and effectively reduce the complexity of the model. Moreover, a lighter and stronger decoder network consisting of deconvolutional layers is designed as a regularization method to suppress overfitting and further improve the performance of DC-CapsNet. We tested our model on the KSC, IN, UP and SA datasets under the condition of a limited number of training samples. The experiments demonstrated that DC-CapsNet achieves a better classification performance than other state-of-the-art models, with a low computation cost. Furthermore, DC-CapsNet has a much simpler architecture than CNN-based models.

Inspired by the potential of DC-CapsNet in the HSI classification field, we will further consider using it to solve HSI processing tasks in other fields in the future, such as target detection and semantic segmentation.

References

[1] C. M. Gevaert, J. Suomalainen, J. Tang, and L. Kooistra, "Generation of Spectral–Temporal Response Surfaces by Combining Multispectral Satellite and Hyperspectral UAV Imagery for Precision Agriculture Applications," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 8, no. 6, pp. 3140-3146, June 2015.

[2] C. Wu, B. Du, and L. Zhang, "Slow Feature Analysis for Change Detection in Multispectral Imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 5, pp. 2858-2874, May 2014.

[3] Y. Naoto, C. Jonathan, and S. Karl, "Potential of Resolution-Enhanced Hyperspectral Data for Mineral Mapping Using Simulated EnMAP and Sentinel-2 Images" R*emote Sens.*, vol. 8, no. 3, p. 172, Feb. 2016.

[4] X. Lu, X. Li, and L. Mou, "Semi-Supervised Multitask Learning for Scene Recognition," *IEEE T. Cybern.*, vol. 45, no. 9, pp. 1967-1976, Sept. 2015.

[5] H. Su, B. Yong, and Q. Du, "Hyperspectral Band Selection Using Improved Firefly Algorithm," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 1, pp. 68-72, Jan. 2016.

[6] H. Su, B. Zhao, Q. Du, P. Du, and Z. Xue, "Multifeature Dictionary Learning for Collaborative Representation Classification of Hyperspectral Imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 4, pp. 2467-2484, April 2018.

[7] H. Su, B. Zhao, Q. Du, and P. Du, "Kernel Collaborative Representation With Local Correlation Features for Hyperspectral Image Classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 2, pp. 1230-1241, Feb. 2019.

[8] H. Su, Y. Yu, Q. Du, and P. Du, "Ensemble Learning for Hyperspectral Image Classification Using Tangent Collaborative Representation," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 6, pp. 3778-3790, June 2020.

[9] Y. Zhao, Y. Qian, and C. Li, "Improved KNN text classification algorithm with MapReduce implementation," in *Proc. Int. Conf. Syst. Informat. (ICSAI)*, Jan. 2018, pp. 1417-1422.

[10] M. A. Bencherif, Y. Bazi, A. Guessoum, N. Alajlan, F. Melgani, and H. AlHichri, "Fusion of Extreme Learning Machine and Graph-Based Optimization Methods for Active Classification of Remote Sensing Images," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 3, pp. 527-531, March 2015.

[11] F. Melgani and L. Bruzzone, "Classification of hyperspectral remote sensing images with support vector machines," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 8, pp. 1778-1790, Aug. 2004.

[12] F. Schmidt, S. Doute, and B. Schmitt, "WAVANGLET: An Efficient Supervised Classifier for Hyperspectral Images," *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 5, pp. 1374-1385, May 2007.

[13] Y. Zhao, L. Zhang, and S. G. Kong, "Band-Subset-Based Clustering and Fusion for Hyperspectral Imagery Classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 2, pp. 747-756, Feb. 2011.

[14] L. Shen and S. Jia, "Three-Dimensional Gabor Wavelets for Pixel-Based Hyperspectral Imagery Classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 12, pp. 5039-5046, Dec. 2011.

[15] 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.

[16] Y. Tarabalka, M. Fauvel, J. Chanussot, and J. A. Benediktsson, "SVM- and MRF-Based Method for Accurate Classification of Hyperspectral Images," *IEEE Geosci. Remote Sens. Lett.*, vol. 7, no. 4, pp. 736-740, Oct. 2010.

[17] 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, June 2014.

[18] 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.

[19] L. Mou, P. Ghamisi, and X. X. Zhu, "Deep Recurrent Neural Networks for Hyperspectral Image Classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 7, pp. 3639-3655, July 2017.

[20] W. Song, S. Li, L. Fang, and T. Lu, "Hyperspectral Image Classification With Deep Feature Fusion Network," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 6, pp. 3173-3184, June 2018.

[21] Y. Chen, K. Zhu, L. Zhu, X. He, P. Ghamisi, and J. A. Benediktsson, "Automatic Design of Convolutional Neural Network for Hyperspectral Image Classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 9, pp. 7048-7066, Sept. 2019.

[22] B. Liu, X. Yu, P. Zhang, X. Tan, A. Yu, and Z. Xue, "A semi-supervised convolutional neural network for hyperspectral image classification," *Remote Sens. Lett.*, vol. 8, no. 9, pp. 839-848, Sept. 2017.

[23] W. Hu, Y. Huang, L. Wei, F. Zhang, and H. Li, "Deep convolutional neural networks for hyperspectral image classification," *J. Sens.*, vol. 2015, no. 2, pp. 1-12, Jul. 2015.

[24] X. Li, M. Ding, and A. Pižurica, "Deep Feature Fusion via Two-Stream Convolutional Neural Network for Hyperspectral Image Classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 4, pp. 2615-2629, April 2020.

[25] A. Santara et al., "BASS Net: Band-Adaptive Spectral-Spatial Feature Learning Neural Network for Hyperspectral Image Classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 9, pp. 5293-5301, Sept. 2017.

[26] B. Pan, Z. Shi, and X. Xu, "R-VCANet: A New Deep-Learning-Based Hyperspectral Image Classification Method," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 10, no. 5, pp. 1975-1986, May 2017.

[27] Y. Chen, H. Jiang, C. Li, X. Jia, and P. Ghamisi, "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.

[28] Y. Li, H. Zhang and Q. Shen, "Spectral–Spatial Classification of Hyperspectral Imagery with 3D Convolutional Neural Network," *Remote Sens.*, vol. 9, no. 1, p. 67, Jan. 2017.

[29] Z. Zhong, J. Li, Z. Luo, and M. Chapman, "Spectral–Spatial Residual Network for Hyperspectral Image Classification: A 3-D Deep Learning Framework," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 2, pp. 847-858, Feb. 2018.

[30] M. E. Paoletti, J. M. Haut, J. Plaza, and A. Plaza, “Deep&Dense Convolutional Neural Network for Hyperspectral Image Classification,” *Remote Sens.*, vol. 10, no. 9, p. 1454, Sep. 2018.
[31] C. Zhang, G. Li, and S. Du, “Multi-Scale Dense Networks for Hyperspectral Remote Sensing Image Classification,” *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 11, pp. 9201-9222, Nov. 2019.
[32] C. Zhang et al., “Deep Feature Aggregation Network for Hyperspectral Remote Sensing Image Classification,” *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 5314-5325, 2020.
[33] C. Mu, Z. Guo, and Y. Liu, “A Multi-Scale and Multi-Level Spectral-Spatial Feature Fusion Network for Hyperspectral Image Classification,” *Remote Sens.*, vol. 12, no. 1, p. 125, Dec, 2019.
[34] Y. Liu, L. Gao, C. Xiao, Y. Qu, K. Zheng, and A. Marinoni, “Hyperspectral Image Classification Based on a Shuffled Group Convolutional Neural Network with Transfer Learning,” *Remote Sens.*, vol. 12, no. 11, p. 1780, May 2020.
[35] H. Liang and Q. Li, “Hyperspectral Imagery Classification Using Sparse Representations of Convolutional Neural Network Features,” *Remote Sens.*, vol. 8, no. 2, p. 99, Dec. 2015.
[36] Z. Gong, P. Zhong, Y. Yu, W. Hu, and S. Li, “A CNN With Multiscale Convolution and Diversified Metric for Hyperspectral Image Classification,” *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 6, pp. 3599-3618, June 2019.
[37] Z. Lu, B. Xu, L. Sun, T. Zhan, and S. Tang, “3-D Channel and Spatial Attention Based Multiscale Spatial–Spectral Residual Network for Hyperspectral Image Classification,” *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 4311-4324, 2020.
[38] L. Zhu, Y. Chen, P. Ghamisi, and J. A. Benediktsson, "Generative Adversarial Networks for Hyperspectral Image Classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 9, pp. 5046-5063, Sept. 2018.
[39] Z. Zhong, J. Li, D. A. Clausi, and A. Wong, "Generative Adversarial Networks and Conditional Random Fields for Hyperspectral Image Classification," *IEEE Trans. Cybern.*, vol. 50, no. 7, pp. 3318-3329, July 2020.
[40] S. Sabour, N. Frosst, and G. E. Hinton, “Dynamic routing between capsules,” 2017, *arXiv:*1710.09829.
[41] S. Jia, B. Zhao, L. Tang, F. Feng, and W. Wang, "Spectral–spatial classification of hyperspectral remote sensing image based on capsule network," *J. Eng.*, vol. 2019, no. 21, pp. 7352-7355, Nov. 2019.
[42] Y. Ma et al., “Classification Based on Capsule Network with Hyperspectral Image,” in *Proc. Int. Geosci. Remote Sens. Symp.*, Nov. 2019, pp. 2750-2753.
[43] F, Deng, S. Pu, X. Chen, Y. Shi, T. Yuan, and S. Pu, “Hyperspectral Image Classification with Capsule Network Using Limited Training Samples.” *Sensors*, vol. 18, no. 9, p. 3153, Sep. 2018.
[44] M. E. Paoletti et al., “Capsule Networks for Hyperspectral Image Classification,” *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 4, pp. 2145-2160, April 2019.
[45] J. Yin, S. Li, H. Zhu, and X. Luo, “Hyperspectral Image Classification Using CapsNet With Well-Initialized Shallow Layers,” *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 7, pp. 1095-1099, July 2019.
[46] R. Lei et al., “A non-local capsule neural network for hyperspectral remote sensing image classification,” *Remote Sens. Lett.*, vol. 12, no. 1, pp. 40-49, Dec. 2020.
[47] H. Li, W. Wang, L. Pan, W. Li, Q. Du and R. Tao, "Robust Capsule Network Based on Maximum Correntropy Criterion for Hyperspectral Image Classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 738-751, 2020.
[48] H. Zhang et al., “1D-Convolutional Capsule Network for Hyperspectral Image Classification,” 2019, *arXiv:*1903.09834.
[49] K. Zhu, Y. Chen, P. Ghamisi, X. Jia, and J. A. Benediktsson, “Deep Convolutional Capsule Network for Hyperspectral Image Spectral and Spectral-Spatial Classification,” *Remote Sens.*, vol. 11, no. 223, p. 582, Jan. 2019.
[50] J. Rajasegaran, V. Jayasundara, S. Jayasekara, H. Jayasekara, S. Seneviratne, and R. Rodrigo, “DeepCaps: Going Deeper With Capsule Networks,” in *Proc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10717-10725.
[51] E. Xi, S. Bing, and Y. Jin, “Capsule Network Performance on Complex Data,” 2017, *arXiv:*1712.03480.
[52] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” 2015, *arXiv:*1502.03167.
[53] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1026-1034.
[54] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, “What is the best multi-stage architecture for object recognition?,” in *Proc. Int. Conf. Comput. Vis.*, May 2010, pp. 2146-2153.
[55] P. A. Lachenbruch and C. J. Lynch, “Assessing screening tests: extensions of McNemar's test.” Stat. Med., vol. 17, no. 19, pp. 2207-2217, Oct. 1998.

Runmin Lei is currently working toward the Master' degree in the School of Civil Engineering, Hefei University of Technology in China. His research interests include hyperspectral remote sensing data analysis and machine learning.

Chunju Zhang is an Associate Professor with the School of Civil Engineering, Hefei University of Technology in China. Her research interests include computational intelligence in remote sensing images and geographic knowledge graph.

Wencong Liu is currently working toward the Master' degree in the School of Civil Engineering, Hefei University of Technology in China. Her research interests include high-resolution image processing and deep learning techniques.

Lei Zhang is currently working toward the Master' degree in the School of Civil Engineering, Hefei University of Technology in China. His research interests include semantic understanding of remote sensing data.

Shihong Du, Institute of Remote Sensing and GIS, Peking University, Beijing, 100871, China.

Xueying Zhang is currently a Professor with Nanjing Normal University, Nanjing, China. Her research interests include intelligent semantic understanding of geographical data including GIS and remote sensing data.