Tackling the over-smoothing problem of CNN-based hyperspectral image classification

Shi He, Huazhu Xue,* Jiehai Cheng, Lei Wang, Yaping Wang, and Yongjuan Zhang

Henan Polytechnic University, School of Surveying and Land Information Engineering, Department of Geographic Information Science, Jiaozuo, China

Abstract. Convolutional neural networks (CNNs) are very important deep neural networks for analyzing visual imagery. However, most CNN-based methods have the problem of over-smoothing at boundaries, which is unfavorable for hyperspectral image classification. To address this problem, a spectral-spatial multiscale residual network (SSMRN) by fusing two separate deep spectral features and deep spatial features is proposed to significantly reduce over-smoothing and effectively learn the features of objects. In the implementation of the SSMRN, a multiscale residual convolutional neural network is proposed as a spatial feature extractor and a band grouping-based bi-directional gated recurrent unit is utilized as a spectral feature extractor. Considering that the importance of spectral and spatial features may vary depending on the spatial resolution of images, we combine both features with two weighting factors with different initial values that can be adaptively adjusted during the network training. To evaluate the effectiveness of the SSMRN, extensive experiments are conducted on public benchmark data sets. The proposed method can retain the detailed boundary of different objects and yield competitive results compared with several state-of-the-art methods. [DOI: 10.1117/1.JRS.16.048506]

Keywords: hyperspectral image classification; over-smoothing; convolutional neural network; residual block; recurrent neural network.

Paper 220258G received Apr. 28, 2022; accepted for publication Nov. 15, 2022; published online Nov. 26, 2022.

1 Introduction

With the rapid development of remote sensing imaging spectroscopy technology, hyperspectral images (HSIs) have become increasingly important in Earth observation due to their rich spectral and spatial information. Classification is an important technique for HSI data exploitation. HSI classification (HSIC) is the task of identifying the category for each pixel with a proper land-cover label,[1] which is more challenging because of the large dimensionality, spectral heterogeneity, and complex spatial distribution of the objects.[2]

To alleviate these problems, traditional HSIC methods involve two steps: (1) feature selection and extraction.[3] This step relies on utilizing feature engineering skills and domain expertise to design several human-engineered features. (2) Classifier training. A classifier in machine learning is an algorithm that automatically orders or categorizes data into one or more of a set of classes. However, the traditional HSIC approaches use handcrafted features to train the classifier. These features may be insubstantial in the case of real data. Therefore, it is difficult to fine-tune between robustness and discriminability as a set of optimal features considerably vary between different data.[4]

Deep neural networks (DNNs) can automatically learn the features from data in a hierarchical manner to construct a model with growing semantic layers until a suitable representation is achieved.[5] To overcome the issue of high intraclass variability and high interclass similarity in HSI, stacked autoencoders[6–8] and deep belief networks[9,10] are introduced as accurate unsupervised methods to extract layerwise trained deep features. However, their standard fully

*Address all correspondence to Huazhu Xue, xhz@hpu.edu.cn

connected (FC) architecture imposes a feature flattening process before the classification, leading to the loss of spatial-contextual information.[11] On the contrary, convolutional neural networks (CNNs) can automatically extract spectral-spatial features from the raw input data. Recurrent neural networks (RNNs) process the spectral information of HSI data as a time sequence considering the spectral bands as time steps. There are three basic models of RNN: (1) Vanilla, (2) long–short-term memory (LSTM), and (3) gated recurrent unit (GRU). Therefore, a large number of CNN or RNN-based methods are proposed for end-to-end modeling and can handle HSI data in spectral and spatial domains individually, and also in a coupled fashion.[12]

For instance, Yang et al.[13] designed a CNN model with two-branch architecture to learn the spectral features and spatial features jointly. Zhong et al.[14] raised an end-to-end three-dimensional (3D) residual CNN architecture for spectral-spatial feature learning and classification. Motivated by the attention mechanism of the human visual system, a residual spectral-spatial attention network (RSSAN)[15] was proposed for HSI classification. To reduce computations, fully convolutional networks were proposed for HSIC.[16] For correctly discovering the contextual relations among pixels, the graph convolutional network was adopted for dealing with the HSIC, which was originally designed for arbitrarily structured non-Euclidean data.[17] The morphological operations, i.e., erosion and dilation, are powerful nonlinear feature transformations. Inspired by these, an end-to-end morphological CNN (MorphCNN)[11] was introduced for HSIC by concatenating the outputs from spectral and spatial morphological blocks extracted in a dual-path fashion. To represent high-level semantic features well, a spectral-spatial feature tokenization transformer (SSFTT) method[18] was proposed to capture spectral-spatial features and high-level semantic features. Keeping in view the sequential property of HSI to determine the class labels, an RNN-based HSIC framework with a novel activation function (parametric rectified tanh) and GRU was proposed.[19] The work[20] proposed a spectral-spatial LSTM-based network that learns spectral and spatial features of HSI by utilizing two separate LSTM-followed Softmax layers for classification, while a decision fusion strategy is implemented to get joint spectral-spatial classification results. In the literature, several works have proposed a CNN joint RNN architecture for HSIC. Spatial-spectral unified network (SSUN) combined a spectral dimensional band grouping-based LSTM model with 2D CNN for spatial features and integrated the spectral feature extraction (FE), spatial FE, and classifier training into a unified neural network.[2] In a spectral-spatial attention network (SSAN),[21] RNN with attention can learn inner spectral correlations within a continuous spectrum, while CNN with attention is designed to focus on saliency features and spatial relevance between neighboring pixels in the spatial dimension. The work[22] integrated CNN with bidirectional convolutional LSTM (CLSTM) in which a 3D CNN model is used to capture low-level spectral-spatial features and CLSTM recurrently analyzes this low-level spectral-spatial information.

CNN is commonly applied to analyze visual imagery.[23] Most of the above methods are based on the CNN backbone and its variants. However, most CNN-based methods have the problem of over-smoothing at boundaries, which is unfavorable for HSIC. DNNs usually yield overfitting methods[24] and are sensitive to perturbations.[25] A large number of training samples are usually required for deep learning methods.[26,27] To significantly reduce the over-smoothing effect and effectively learn the features of objects, a multi-task learning spectral-spatial multiscale residual network (SSMRN) is proposed for the end-to-end HSIC. The contributions can be summarized as follows:

1. An end-to-end SSMRN is designed by fusing two separate deep spectral features and deep spatial features to extract spectral-spatial features for HSIC. The model yields competitive results under different training sample conditions.
2. The proposed framework takes the weight between spectral and spatial features into consideration, which increases the influence of the current pixel and reduces over-smoothing. Meanwhile, the multi-task learning technology is integrated into the framework, improving the stability of results.

The rest of the sections are organized as follows. First, Sec. 2 introduces the preliminary knowledge of CNN, residual networks, and RNN. The proposed architecture along with the design methodology is introduced in Sec. 3. Next, experimental data sets and results are given

in Sec. 4. Then, the impact of the SSMRN architecture on classification results is analyzed in Sec. 5. Finally, Sec. 6 concludes the paper with a summary of the proposed method and the scope of future work.

2 Preliminary

In this section, we mainly recall the background information on CNN, residual networks, and RNN.

2.1 *Convolutional Neural Network*

A CNN[28] is a class of DNNs, most commonly applied to analyzing visual imagery. Three main types of layers are used to build CNNs architectures: convolutional layer, pooling layer, and FC layer. Compared with multilayer perceptron neural networks, CNNs are easier to train because of the parameter sharing scheme and local connectivity.

While CNN-based methods have achieved large improvement in HSIC, they usually suffer from severe over-smoothing problems at edge boundaries. There are two major reasons: (1) the scales between supervised information and spatial features do not match. The supervised information of HSIC is pixel-level, while the spatial features are extracted from the neighbourhood of the current pixel. (2) The parameter sharing scheme makes the spatial features extracted for the patch instead of the current pixel. Two major reasons lead to an insufficient influence of the current pixel in classification. Attentional mechanisms can counteract the effects of parameter sharing,[15,21] but increase the amount of computation. A smaller size patch will also decrease the possibility of the over-smoothing phenomenon[2] but result in insufficient extraction of spatial information and lower classification accuracy (CA).[29] Another approach is to utilize superpixel segmentation,[17] but the segmentation algorithm affects the classification results.

2.2 *Residual Networks*

A residual network is an effective extension to CNNs that has empirically shown to increase performance in ImageNet classification. A residual network does this by utilizing a skip connection to jump over some layers. As shown in Fig. 1, the typical residual block is implemented with double-layer skips that contain nonlinearities. The skip connections between layers add the outputs from previous layers to the outputs of stacked layers. One motivation for skipping over layers is to avoid the problem of vanishing gradients, by reusing activations from a previous layer until the adjacent layer learns its weights.[30] Skipping effectively simplifies the network, using fewer layers in the initial training stages. The residual block is easy to understand and optimize and can be stacked to any depth and embedded in any existing CNN.

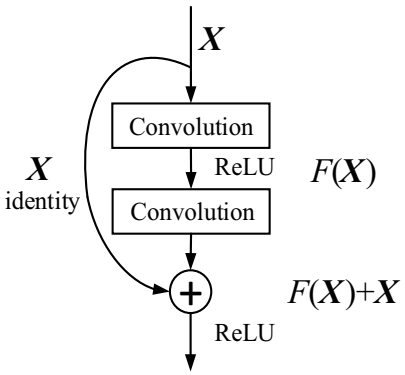

Fig. 1 Architecture of a typical residual block.

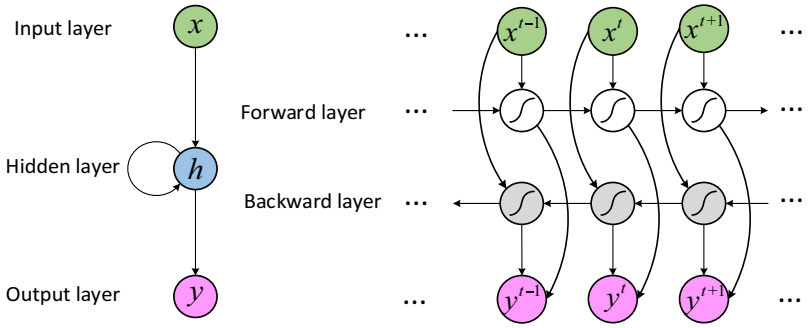

Fig. 2 Architecture of Bi-RNN.

2.3 *Recurrent Neural Network*

RNNs allow us to operate over sequences of input, output, or both at the same time. RNN makes them applicable to challenging tasks involving sequential data such as speech recognition and language modeling. LSTM and GRU[31] are introduced to learn long-term dependencies and alleviate the vanishing/exploding gradient problem. These two architectures do not have any fundamental differences from each other, but they use different functions to compute the hidden state. LSTM is strictly stronger than the GRU as it can easily perform unbounded counting. The GRU has fewer parameters than LSTM, and GRU has been shown to exhibit better performance on certain smaller and less frequent data sets. Bi-directional RNNs (Bi-RNNs) utilize a finite sequence to predict or label each element of the sequence based on the element's past and future contexts, as shown in Fig. 2. Bi-RNN concatenating the outputs of two RNNs allows them to receive information from the sequence from left to right, the other one from right to left.

Hyperspectral data usually have hundreds of bands. So, pixel classification in HSI can be treated as a many-to-one task where we are given a sequence of bands of a pixel and then classify what classification that pixel is. A natural idea is to consider each band as a time step. The large length of RNNs input sequence can lead to an overfitting issue, which consumes high computing and storage resources. In addition, a large number of spectral channels and limited training samples restrict the performance of HSIC.[26]

3 Proposed Framework

The deep networks used for HSIC are divided into spectral-feature networks, spatial-feature networks, and spectral-spatial-feature networks. To effectively learn the features of objects, we utilize the spectral-spatial-feature networks to extract joint deep spectral-spatial features for HSIC. The joint deep spectral-spatial features are mainly obtained by the following three ways:[32] (1) mapping the low-level spectral-spatial features to high-level spectral-spatial features via deep networks; (2) directly extracting deep features from original data or several principal components of the original data; and (3) fusing two separate deep spectral features and deep spatial features. Considering that the importance of spectral and spatial features may vary depending on the spatial resolution of images, we adopt the way of fusing two separate deep features to conveniently adjust the influence of different features on the classification results.

Three sections are playing crucial roles in our methodology: a multiscale residual CNN (MRCNN)-based spatial feature learner, a bi-directional GRU (bi-GRU)-based spectral feature learner, and a multi-task learning model that combines both features with two weighting factors.

3.1 *Multiscale Residual CNN for Spatial Classification*

The proposed MRCNN architecture is shown in Fig. 3. Let $\boldsymbol{X} \in \mathbb{R}^{r\times c\times b}$ be the original HSI data, where r, c and b are the row number, column number, and band number, respectively. First, to suppress noise and reduce the computational costs, the principal component analysis is applied to the original HSI data, and only the first p principle components are reserved. Denote the dimension-reduced data by $\boldsymbol{X}_p \in \mathbb{R}^{r\times c\times p}$. Around each pixel, a neighbor region is extracted with the size of $k \times k \times p$ as the input of the spatial branch.

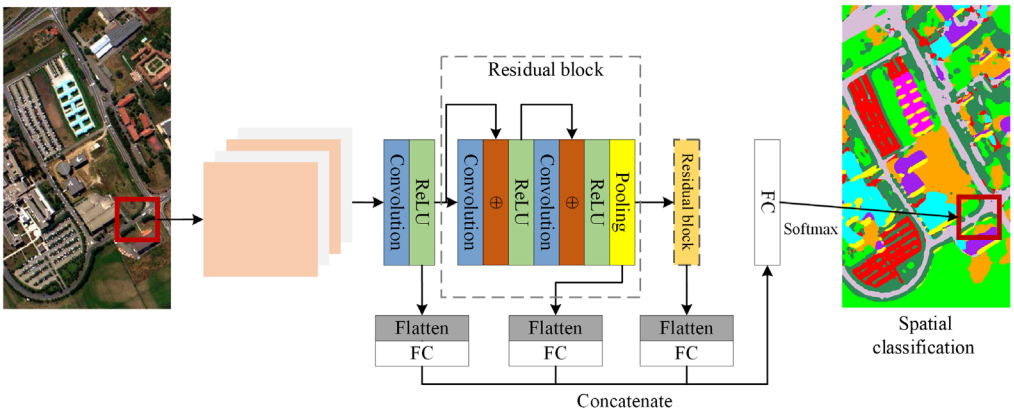

Fig. 3 Architecture of the proposed MRCNN.

Considering the complex environment of the HSI, where different objects tend to have different scales, we propose to extract both shallow and deep features by applying a convolution layer with rectified linear unit (ReLU) activation and two residual blocks in the classification. The local max pooling layer is adopted in residual blocks. We add a flatten layer and an FC layer with the same number of neurons after each scale output. Then, these FC layers are merged into a new FC layer. Let $\boldsymbol{h}^{(j)} = f(\boldsymbol{W}^{(j)}\boldsymbol{x}^{(j)} + \boldsymbol{b}^{(j)})$, $j = 1,2,3$ denotes the j'th FC layer, where $\boldsymbol{x}^{(j)}$ is the flattened features in the jth flatten layer, $\boldsymbol{W}^{(j)}$ and $\boldsymbol{b}^{(j)}$ are the corresponding weight matrix and bias term, respectively. The fourth FC layer $\boldsymbol{h}^{(4)}$ can be calculated as $\boldsymbol{h}^{(4)} = \text{concat}[\boldsymbol{h}^{(1)}, \boldsymbol{h}^{(2)}, \boldsymbol{h}^{(3)}]$. In this way, features in different layers are taken into consideration during the classification stage, and the network will possess the property of multiscale.

The loss function for cross entropy of MRCNN can be expressed as

$$\mathcal{L} = -\frac{1}{M}\sum_{m=1}^{M}\sum_{n=1}^{N} y_m^n \log(\hat{y}_m^n), \tag{1}$$

where y_m^n and $\hat{y}_m^n$ denote the truth and predicted labels, respectively. M is the number of training samples and N is the number of classes.

3.2 *Bi-GRU for Spectral Classification*

GRU has fewer parameters than LSTM for modeling various sequential problems, and Bi-GRU allows the sequential vector to be fed into the architecture one by one to learn continuous features with forward and backward directions. So, we utilize Bi-GRU for spectral classification.

The complete spectral classification framework is shown in Fig. 4. To reduce computation, a suitable grouping strategy[2] is used in this paper. For each pixel $\boldsymbol{x}$ in the HSI, let $\boldsymbol{x} = (\lambda_1, \lambda_2, \ldots, \lambda_j, \ldots \lambda_b)^{\mathrm{T}}$ be the spectral vector, where λ_j is the reflectance of the j'th band and b is the number of bands. Let $r(\ll b)$ be the number of time steps (e.g., number of groups). The transformed sequences can be denoted by $(\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_t, \ldots \boldsymbol{c}_r)$, where $\boldsymbol{c}_t$ is the sequence at the tth time step. Specifically, the grouping strategy is

$$\begin{aligned}
\boldsymbol{c}_1 &= (\lambda_1, \lambda_{1+r}, \ldots, \lambda_{1+(m-1)r})^{\mathrm{T}} \\
\boldsymbol{c}_2 &= (\lambda_2, \lambda_{2+r}, \ldots, \lambda_{2+(m-1)r})^{\mathrm{T}} \\
&\ldots \\
\boldsymbol{c}_t &= (\lambda_t, \lambda_{t+r}, \ldots, \lambda_{t+(m-1)r})^{\mathrm{T}} \\
&\ldots \\
\boldsymbol{c}_r &= (\lambda_r, \lambda_{r+r}, \ldots, \lambda_{r+(m-1)r})^{\mathrm{T}},
\end{aligned} \tag{2}$$

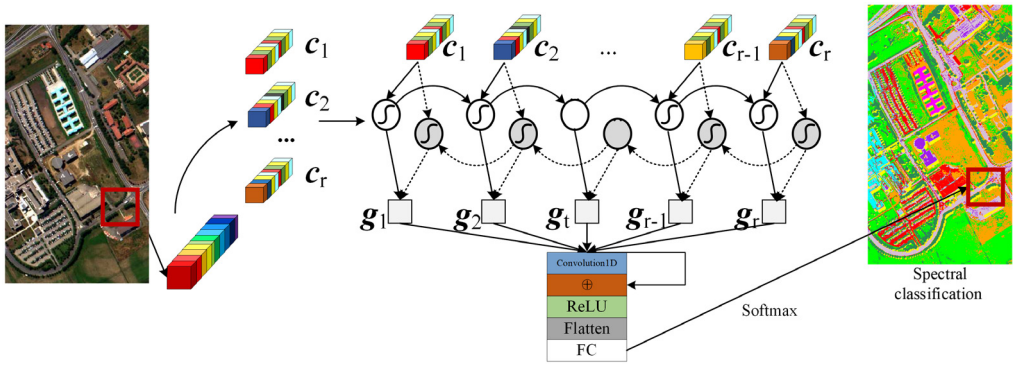

Fig. 4 Band grouping-based Bi-GRU model for spectral classification.

where $m = \text{floor}(b/r)$ is the sequence length of each time step and floor$(\cdot)$ function rounds numbers down. After grouping, spectral vector $\boldsymbol{x}$ is transformed into sequences $(\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_t, \ldots \boldsymbol{c}_r)$.

The input to our model is the sequences $(\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_t, \ldots \boldsymbol{c}_r)$, and the bi-directional hidden vector is calculated as

Forward hidden state:

$$\boldsymbol{h}_t^{(1)} = \tanh(\boldsymbol{W}^{(1)} \cdot \boldsymbol{c}_t + \boldsymbol{U}^{(1)} \cdot \boldsymbol{h}_{t-1}^{(1)} + \boldsymbol{b}^{(1)}). \tag{3}$$

Backward hidden state:

$$\boldsymbol{h}_t^{(2)} = \tanh(\boldsymbol{W}^{(2)} \cdot \boldsymbol{c}_t + \boldsymbol{U}^{(2)} \cdot \boldsymbol{h}_{t+1}^{(2)} + \boldsymbol{b}^{(2)}), \tag{4}$$

where the coefficient matrices $\boldsymbol{W}^{(1)}$ and $\boldsymbol{W}^{(2)}$ are from the input at the present step, $\boldsymbol{U}^{(1)}$ is from the hidden state $\boldsymbol{h}_{t-1}^{(1)}$ at the previous step, $\boldsymbol{U}^{(2)}$ is from $\boldsymbol{h}_{t+1}^{(2)}$ at the succeeding step, tanh is the hyperbolic tangent, and the memory of the input as the output of this encoder is $\boldsymbol{g}_t$:

$$\boldsymbol{g}_t = \text{concat}(\boldsymbol{h}_t^{(1)}, \boldsymbol{h}_t^{(2)}), \tag{5}$$

where concat$(\cdot)$ is a function of concatenation between the forward hidden state and backward hidden state.

The grouping strategy uses the original HSI spectral vector as the feature of the new sequence and the RNN uses the parameter sharing scheme, so a one-dimensional convolutional residual block is added to reassign the weight of the feature based on the channel attention mechanism. So, we can compute the predicted label y_i of pixel $\boldsymbol{x}_i$ as follows:

$$y_i = V(\text{F}_{1d}(\boldsymbol{g}_1, \ldots, \boldsymbol{g}_t, \ldots, \boldsymbol{g}_r) + (\boldsymbol{g}_1, \ldots, \boldsymbol{g}_t, \ldots, \boldsymbol{g}_r)), \tag{6}$$

where $\text{F}_{1d}(\cdot)$is one-dimensional convolutional layer with stride one and $V(\cdot)$ indicates a series of operations as shown in Fig. 4, including a ReLU activation, a flatten function, an FC layer and a Softmax activation function.

3.3 *SSMRN*

The proposed SSMRN framework is shown in Fig. 5, which starts with two branches, learning the spatial and spectral features, respectively. Then, concatenate these two branches into a layer. λ_{spatial} and $\lambda_{\text{spectral}}$ are the corresponding weighting factors.

To better train the whole network, two auxiliary tasks are added to the framework.[2] So, the proposed SSMRN is a triple-task framework, including one main task (classification based on spectral-spatial information) and two auxiliary tasks (classification based on spectral information and classification based on spatial information). The complete loss function for cross entropy of the SSMRN is defined as

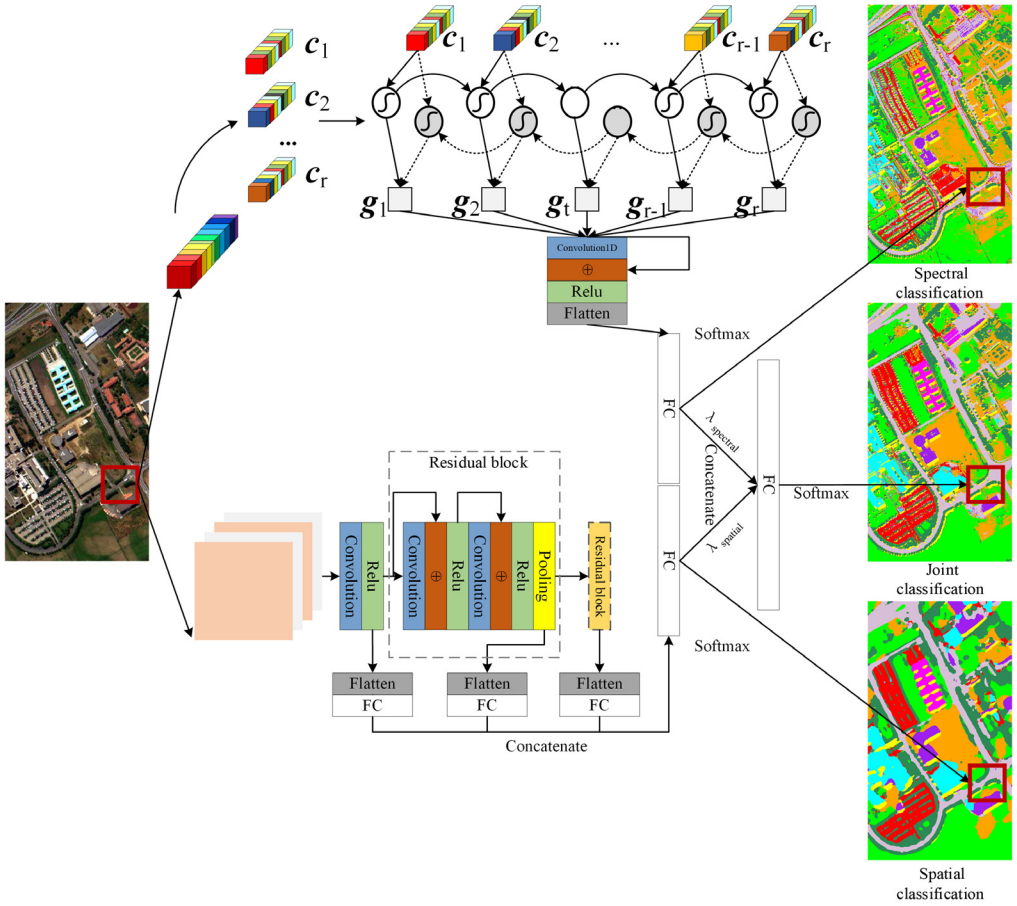

Fig. 5 Architecture of the proposed SSMRN.

$$\mathcal{L} = \mathcal{L}^{\text{joint}} + \mathcal{L}^{\text{spectral}} + \mathcal{L}^{\text{spatial}} = -\frac{1}{M}\sum_{m=1}^{M}\sum_{n=1}^{N} y_m^n \log(\hat{y}_m^{n\text{joint}}) - \frac{1}{M}\sum_{m=1}^{M}\sum_{n=1}^{N} y_m^n \log(\hat{y}_m^{n\text{spectral}}) - \frac{1}{M}\sum_{m=1}^{M}\sum_{n=1}^{N} y_m^n \log(\hat{y}_m^{n\text{spatial}}), \tag{7}$$

where $\mathcal{L}^{\text{joint}}$ is the main loss function, $\mathcal{L}^{\text{spectral}}$ and $\mathcal{L}^{\text{spatial}}$ are two auxiliary loss functions,$\hat{y}_m^{n\text{joint}}$, $\hat{y}_m^{n\text{spectral}}$, and $\hat{y}_m^{n\text{spatial}}$ are the corresponding predicted labels, y_m^n is the true label. M is the number of training samples and N is the number of classes. The whole network is trained in an end-to-end manner, where all the parameters are optimized by the batch stochastic gradient descent algorithm at the same time. In this way, the complete loss function will balance the convergences of both the whole network and the subnetworks.

4 Experiment

In this section, we introduce three public data sets used in our experiment and the configuration of the proposed SSMRN. In addition, classification performance based on the proposed method and other comparative methods is presented.

4.1 *Experimental Data*

Three publicly available hyperspectral data sets are utilized to evaluate the performance of the proposed method, i.e., Indian Pines (IP) from the airborne visible/infrared imaging spectrometer

Table 1 Summary of the HSI datasets used for experimental evaluation.

Data set	Source	Number of pixels	Number of spectral reflectance bands	Wavelength range (10^{-6} m)	Spatial resolution (m/pixel)	Number of classes
IP	AVIRIS	145×145	200	0.4–2.5	20	16
PU	ROSIS	613×340	103	0.43–0.86	1.3	9
SA	AVIRIS	512×217	204	0.4–2.5	3.7	16

(AVIRIS) sensor, Pavia University (PU) from the reflective optics systems imaging spectrometer (ROSIS) sensor, and Salinas (SA) from the AVIRIS sensor. The data set details are shown in the following Table 1.

4.2 *Experimental Setting*

4.2.1 *Evaluation indicators*

To quantitatively analyze the effectiveness of the proposed method and other methods for comparison, three quantitative evaluation indexes are introduced, including class-specified CA, overall classification accuracy (OA), and Kappa coefficient (Kappa). The larger value of each indicator represents a better classification effect.

4.2.2 *Configuration*

All the experiments are implemented with an Intel(R) Xeon(R) Sliver 4210 CPU @ 2.20-GHz with 64 GB of RAM and an NVIDIA RTX2080 graphic card, TensorFlow 2.3.1, and Keras 2.4.3 with python 3.7.6. We use the Adam optimizer to train the networks with a learning rate of 0.001. The gradient of each weight is individually clipped so that its norm is no higher than 1. The training epochs are set as 1500 with batch size 1048.

4.2.3 *Parameter setting*

All the experiments in this paper are randomly repeated 30 times. In each repetition, we first randomly generate the training set from the whole data set with the same number of the labelled class. Then, the remaining samples make up the test set. Details are given in Tables 2–4.

For the proposed MRCNN, the input is a $24 \times 24 \times 4$ patch, where 4 is the number of reserved principal components. All convolutional layers have 64 filters. The kernel size of the first left convolutional layer is 1×1, and the other kernel sizes are 3×3. The size of the max pooling layers is 2×2. The three FC layers after each scale output each own 64 units. For the proposed Bi-GRU, let 3 be the number of time steps. The hidden size in GRU is 64, so one-dimensional convolutional layers have 128 filters because of Bi-GRU. For the proposed SSMRN, the input is as same as the Bi-GRU and MRCNN. The number of neurons of the FC layer in the spectral branch and spatial branch is 192, so the number of neurons in the joint FC layer is 384.

In our study, we adopt the way of fusing two separate deep spectral features and deep spatial features. Since the importance of spectral and spatial features may vary depending on different spatial resolutions, we consider the weight of these two parts and we need to specify the initial value of these hyperparameters. The principle is that the higher the spatial resolution and the smaller the influence of the mixed pixel effect, the greater the initial spectral weight should be. Suppose the sum of the two weights is 1 and the weights for both parts are close to each other. Owing to the proposed strategy, the weights for the spectral and spatial parts can be adjusted adaptively. The initial value of weighting factors λ_{spatial} and $\lambda_{\text{spectral}}$ are given in Table 5.

Table 2 Number of training and test samples used in the IP data set.

	Class name	Training	Test
1	Alfalfa	30	16
2	Corn-notill	30	1398
3	Corn-mintill	30	800
4	Corn	30	207
5	Grass-pasture	30	453
6	Grass-trees	30	700
7	Grass-pasture-mowed	15	13
8	Hay-windrowed	30	448
9	Oats	15	5
10	Soybean-notill	30	947
11	Soybean-mintill	30	2425
12	Soybean-clean	30	563
13	Wheat	30	175
14	Woods	30	1235
15	Buildings-grass-trees-drives	30	356
16	Stone-steel-towers	30	63

Table 3 Number of training and test samples used in the PU data set.

	Class name	Training	Test
1	Asphalt	200	6431
2	Meadows	200	18,449
3	Gravel	200	1899
4	Trees	200	2864
5	Painted metal sheets	200	1145
6	Bare soil	200	4829
7	Bitumen	200	1130
8	Self-blocking bricks	200	3482
9	Shadows	200	747

4.2.4 *Ablation study*

In this section, we compare the SSMRN with the SSMRN without auxiliary tasks. As shown in Table 6, the SSMRN surpasses the SSMRN without auxiliary tasks, especially for small samples of the IP data set. These results demonstrate that multi-task learning can select the useful HSI data for feature learning.

4.3 *Classification Results*

To demonstrate the superiority and effectiveness of the proposed SSMRN model, it is compared with the proposed Bi-GRU, MRCNN, and advanced spectral-spatial DNNs methods, such as

Table 4 Number of training and test samples used in the SA data set.

	Class name	Training	Test
1	Brocoli_green_weeds_1	200	1809
2	Brocoli_green_weeds_2	200	3526
3	Fallow	200	1776
4	Fallow_rough_plow	200	1194
5	Fallow_smooth	200	2478
6	Stubble	200	3759
7	Celery	200	3379
8	Grapes_untrained	200	11,071
9	Soil_vinyard_develop	200	6003
10	Corn_senesced_green_weeds	200	3078
11	Lettuce_romaine_4wk	200	868
12	Lettuce_romaine_5wk	200	1727
13	Lettuce_romaine_6wk	200	716
14	Lettuce_romaine_7wk	200	870
15	Vinyard_untrained	200	7068
16	Vinyard_vertical_trellis	200	1607

Table 5 Initial value of weighting factors.

	Data set	Spatial resolution (m/pixel)	$\lambda_{spatial}$	$\lambda_{spectral}$
1	IP	20	0.5	0.5
2	PU	1.3	0.4	0.6
3	SA	3.7	0.4	0.6

Table 6 OA (%) of SSMRN with two modules.

Module	IP	PU	SA
SSMRN	94.87 ± 1.21	99.90 ± 0.07	99.76 ± 0.14
SSMRN without auxiliary tasks	92.57 ± 1.26	99.77 ± 0.17	99.71 ± 0.24

SSUN,[2] SSAN,[21] RSSAN,[15] MorphCNN,[11] and SSFTT.[18] Bi-GRU is the spectral FE branch of SSMRN. MRCNN is the spatial FE branch of SSMRN. SSUN, SSAN, RSSAN, MorphCNN, SSFTT, and SSMRN are all based on the CNN backbone and its variants, integrating spatial features, and spectral features. RSSAN and SSFTT directly extract the joint deep spectral-spatial features via CNNs. SSUN, SSAN, MorphCNN, SSFTT, and SSMRN obtain deep spectral features and deep spatial features via two deep networks. And the two kinds of features are fused to generate the joint deep spectral-spatial features. The difference is that SSMRN considers the weight relationship between the spectral and spatial branches depending on the spatial resolution of images, and embeds multi-task learning technology at the same time.

For SSUN, SSAN, and MorphCNN, the input is a $24 \times 24 \times 4$ patch, where 4 is the number of reserved principal components. Limited by our computer configuration, we cannot run RSSAN properly with the original input size in the corresponding reference, so the input of RSSAN is a $24 \times 24 \times 8$ patch, where 8 is the number of reserved principal components instead of the number of spectral bands. According to the reference, the input of SSFTT is a $13 \times 13 \times 30$ patch. For the SSUN, SSAN, RSSAN, MorphCNN, and SSFTT, all network settings are as described in their corresponding references. For a fair comparison, the training sample sets and test sample sets of all methods are randomly selected, as shown in Tables 2–4.

Quantitative evaluation: Tables 7–9 report the CA, OA, and Kappa using all the mentioned methods for the IP, PU, and SA datasets, respectively. All algorithms are executed 30 times. The average results with the standard deviation obtained are reported to reduce random selection effects. The optimal results are denoted in bold. The evaluation data clearly show that the proposed SSMRN method performs the best. The SSMRN obtains the highest OA and Kappa. SSMRN also generates most of the highest class-specific accuracy, where the results of a few classes have slightly lower precisions than MRCNN, SSUN, and SSFTT. Particularly in the IP datasets, the results of SSMRN are higher than other methods, which shows that SSMRN can

Table 7 Classification results of different methods for the IP data set. Bold indicates the best result.

Label	Class name	Bi-GRU	MRCNN	SSUN	SSAN	RSSAN	MorphCNN	SSFTT	SSMRN
1	Alfalfa	95.62	**100**	**100**	**100**	99.79	96.25	**100**	99.79
2	Corn-notill	68.39	81.13	72.35	78.24	63.24	78.16	84.76	**87.76**
3	Corn-mintill	64.40	93.60	88.92	89.47	74.40	82.86	90.21	**95.75**
4	Corn	82.25	98.96	96.71	98.24	91.75	92.75	99.14	**99.35**
5	Grass-pasture	89.25	95.65	93.00	92.81	89.83	85.24	94.77	**96.57**
6	Grass-trees	94.90	97.24	92.85	94.90	94.14	87.23	99.10	**99.47**
7	Grass-pasture-mowed	96.41	**100**	99.74	**100**	**100**	93.33	**100**	**100**
8	Hay-windrowed	96.46	99.88	99.69	99.61	99.63	97.66	99.90	**99.99**
9	Oats	99.33	**100**	**100**	**100**	**100**	91.33	**100**	**100**
10	Soybean-notill	77.99	90.99	79.67	82.02	76.68	87.54	89.97	**94.29**
11	Soybean-mintill	59.77	88.91	75.08	82.37	70.79	88.70	84.42	**92.11**
12	Soybean-clean	78.65	91.36	88.08	88.88	77.82	81.52	87.75	**96.52**
13	Wheat	99.12	99.88	99.37	99.94	98.93	91.90	99.90	**99.92**
14	Woods	83.94	97.86	89.92	93.80	87.82	94.33	97.02	**99.03**
15	Buildings-grass-trees-drives	76.89	98.68	96.65	98.26	91.89	94.44	98.56	**99.89**
16	Stone-steel-towers	94.55	99.31	99.78	99.47	98.46	97.30	99.31	**100**
	OA (%)	74.96	91.93	84.02	87.70	78.98	87.48	90.74	**94.87**
		±1.22	±2.05	±1.55	±1.41	±2.29	±13.86	±1.41	±1.21
	Kappa ×100	71.73	90.88	81.89	86.02	76.22	85.58	89.46	**94.13**
		±1.34	±2.23	±1.73	±1.57	±2.56	±16.22	±1.59	±1.38
	Runtime (s)	59.09	111.66	50.75	162.26	89.97	205.27	58.68	160.58
		±0.75	±0.57	±0.59	±1.28	±0.47	±1.84	±0.12	±0.84

Table 8 Classification results of different methods for the PU data set. Bold indicates the best result.

Label	Class name	Bi-GRU	MRCNN	SSUN	SSAN	RSSAN	MorphCNN	SSFTT	SSMRN
1	Asphalt	88.74	99.74	95.92	96.86	98.68	91.55	99.39	**99.81**
2	Meadows	92.26	99.88	97.29	96.53	99.52	83.82	99.76	**99.94**
3	Gravel	86.05	99.93	97.39	98.64	99.31	92.32	99.45	**99.97**
4	Trees	96.84	99.26	99.29	98.28	98.63	91.92	98.88	**99.61**
5	Painted metal sheets	99.73	99.82	**99.98**	99.97	99.82	96.98	99.90	99.96
6	Bare soil	93.68	98.03	99.00	99.63	99.87	90.45	**99.99**	**99.99**
7	Bitumen	93.79	**100**	99.09	99.77	99.74	89.42	99.99	99.99
8	Self-blocking bricks	85.45	99.64	98.39	98.61	98.47	92.58	98.39	**99.85**
9	Shadows	99.85	99.45	99.83	99.92	99.38	95.37	99.51	**99.99**
	OA (%)	91.71	99.57	97.68	97.59	99.28	88.25	99.54	**99.90**
		±0.64	±1.31	±0.67	±1.20	±0.37	±23.50	±0.17	±0.07
	Kappa ×100	89.00	99.41	96.90	96.79	99.03	86.51	99.38	**99.87**
		±0.82	±1.82	±0.89	±1.59	±0.50	±24.85	±0.23	±0.10
	Runtime (s)	107.75	332.85	98.55	321.25	335.54	744.56	204	394.69
		±1.61	±4.81	±0.92	±2.00	±3.94	±5.14	±1.89	±5.15

effectively learn the features of objects, especially under the condition of a small number of samples. The CA, OA, and Kappa of Bi-GRU are lower than other methods, specifically in Table 5. Because Bi-GRU only uses the spectral feature, and the IP datasets have lower spatial resolution and the bigger influence of the mixed pixel effect. MRCNN's results are second only to SSMRN, which shows that good results can be obtained using spatial features and proper deep network structure. Especially in the SA data set, the results of the MRCNN and SSMRN models are almost identical. The likely reason is that the ground objects of interest in the image are homogeneous, regular, and have a large area. The pixel-level supervised information can be better regarded as the patch-level supervised information. The scales between supervised information and spatial features match. The structures of SSUN and SSAN are similar to that of SSMRN, which belongs to the way of fusing two separate deep spectral features and deep spatial features. However, the reason why the results of SSUN and SSAN are not as good as SSMRN may be that the network depth of spectral and spatial FE is not enough. The structures of RSSAN, MorphCNN, and SSFTT belong to the way of directly extracting deep features from original data or several principal components of the original data. The RSSAN and SSFTT are powerful methods. The main limitation of RSSAN and SSFTT is that a certain number of samples are required, which may result in poor performance with small samples, such as in the IP data sets. The accuracy of classification results of MorphCNN is low and unstable in Tables 7 and 8. Because compared with the objects in PU and SA, the morphological feature contained in the patch is not obvious.

As shown in Tables 7–9, Bi-GRU, SSUN, and SSFTT generally cost less time than MRCNN and other spectral-spatial feature methods. The reasons may be the grouping strategy of Bi-GRU, the grouping strategy and insufficient network depth of SSUN, and the transformer encoder module of SSFTT. The runtime of the MorphCNN is the longest. The reason is that network structure is more complex and deeper than other networks.

Tables 10–12 show the OA of SSUN, SSAN, RSSAN, MorphCNN, SSFTT, and SSMRN with different training samples. Considering the stability and robustness of the proposed method under different training samples, 5, 10, 15, and 30 labeled samples of each class are randomly

Table 9 Classification results of different methods for the SA data set. Bold indicates the best result.

Label	Class name	Bi-GRU	MRCNN	SSUN	SSAN	RSSAN	MorphCNN	SSFTT	SSMRN
1	Brocoli_green_weeds_1	99.45	**100**	99.84	**100**	99.97	92.84	**100**	99.99
2	Brocoli_green_weeds_2	99.85	**100**	99.83	99.96	99.98	82.84	99.99	99.99
3	Fallow	99.70	**100**	99.83	99.99	99.94	72.61	**100**	99.99
4	Fallow_rough_plow	99.47	**99.98**	99.87	99.87	99.93	87.74	99.77	99.92
5	Fallow_smooth	98.90	**99.91**	99.53	99.67	99.90	76.90	99.63	99.54
6	Stubble	99.89	**100**	99.96	99.90	99.97	89.16	99.99	99.99
7	Celery	99.76	**99.98**	99.95	99.95	99.89	87.45	99.96	99.94
8	Grapes_untrained	79.35	99.11	90.17	96.44	96.36	59.83	98.33	**99.39**
9	Soil_vinyard_develop	99.66	**100**	99.85	99.70	99.78	60.71	99.99	99.98
10	Corn_senesced_ green_weeds	95.51	**99.97**	99.08	99.70	99.87	70.71	99.78	99.64
11	Lettuce_romaine_4wk	99.02	99.93	99.92	99.61	99.90	73.04	**100**	99.97
12	Lettuce_romaine_5wk	99.96	**100**	99.96	99.82	99.95	48.45	**100**	**100**
13	Lettuce_romaine_6wk	99.32	**100**	99.96	99.86	99.92	53.79	99.99	99.92
14	Lettuce_romaine_7wk	98.26	**100**	99.93	99.93	99.96	65.59	99.93	99.83
15	Vinyard_untrained	76.38	99.41	95.68	98.32	98.53	46.15	99.18	**99.70**
16	Vinyard_vertical_trellis	99.16	**100**	99.90	99.89	99.82	88.73	99.93	99.95
	OA (%)	91.71	99.71	97.13	98.89	98.94	68.15	99.48	**99.76**
		±0.86	±0.28	±0.48	±0.46	±1.18	±16.64	±0.38	±0.14
	Kappa ×100	90.73	99.68	96.79	98.76	98.82	64.31	99.42	**99.74**
		±0.94	±0.31	±0.54	±0.51	±1.32	±19.06	±0.42	±0.16
	Runtime (s)	192.09	595.64	159.61	1065.16	577.14	1333.84	365.99	740.52
		±1.85	±2.04	±1.09	±8.65	±2.68	±40.29	±0.99	±4.54

Table 10 OA(%) of different methods under different training sample numbers of each class on the IP data set. Bold indicates the best result.

	Training sample numbers of each class	SSUN	SSAN	RSSAN	MorphCNN	SSFTT	SSMRN
1	5	57.93	57.29	50.59	62.22	**67.87**	67.80
2	10	69.88	70.87	62.72	77.73	78.13	**83.91**
3	15	75.13	77.17	68.37	68.20	83.88	**89.23**
4	30	84.02	87.70	78.98	87.48	90.74	**94.87**

selected as training data for the IP and 30, 50, 100, and 200 for the PU and SA in the experiment. With the change in sample size, the results of MorphCNN fluctuate sharply. It further proves that the morphological feature is unstable. As the number of samples increases, the results of SSUN, SSAN, RSSAN, SSFTT, and SSMRN become better. And SSMRN significantly outperforms

Table 11 OAs (%) of different methods under different training sample numbers of each class on the PU data set. Bold indicates the best result.

	Training sample numbers of each class	SSUN	SSAN	RSSAN	MorphCNN	SSFTT	SSMRN
1	30	87.27	84.57	87.84	82.40	94.90	**95.15**
2	50	90.46	88.46	91.98	80.84	97.00	**97.71**
3	100	95.09	94.10	97.75	90.10	98.76	**99.55**
4	200	97.68	97.59	99.28	88.25	99.54	**99.90**

Table 12 OAs (%) of different methods under different training sample numbers of each class on the SA data set. Bold indicates the best result.

	Training sample numbers of each class	SSUN	SSAN	RSSAN	MorphCNN	SSFTT	SSMRN
1	30	93.84	93.80	95.79	80.05	96.42	**97.63**
2	50	95.03	95.46	96.59	88.95	97.65	**98.48**
3	100	96.35	97.38	96.56	77.96	98.86	**99.51**
4	200	97.13	98.89	98.94	68.15	99.48	**99.74**

other methods under different training sample conditions. In the case of a small number of samples, our method still results in a good performance. In addition, when the number of samples of each class is 100 in PU and SA, the OAs of all the other methods are <99%, but the accuracy of the SSMRN can reach 99.5%. These prove that SSMRN can effectively learn the features of objects under different training sample conditions.

Qualitative evaluation: the classification maps of different methods are shown in Figs. 6–8. By visual comparison, the classification map obtained by SSMRN is the cleanest and the closest

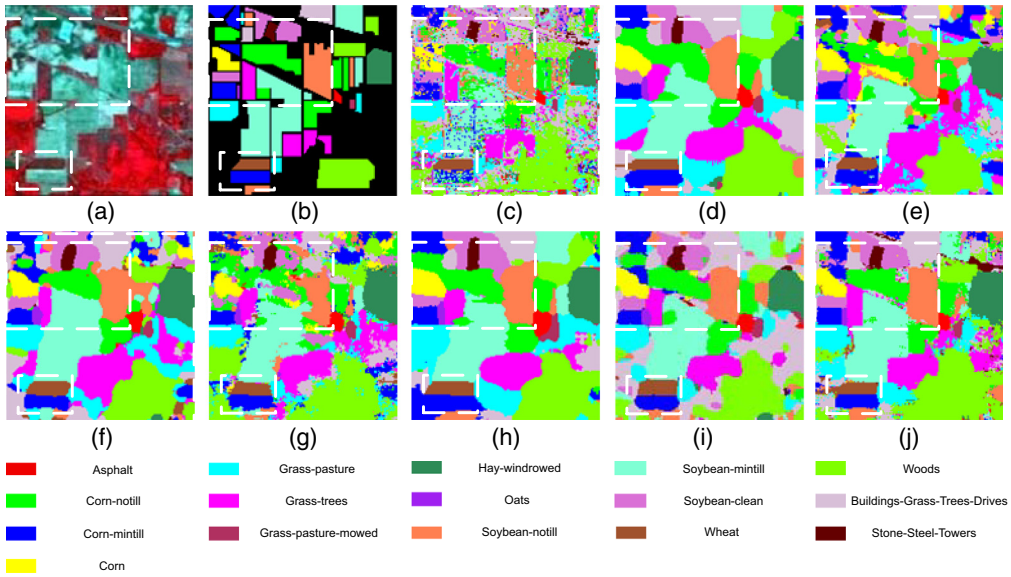

Fig. 6 IP data set and classification maps using different methods: (a) false-color image; (b) ground-truth map; (c) Bi-GRU; (d) MRCNN; (e) SSUN; (f) SSAN; (g) RSSAN; (h) MorphCNN; (i) SSFTT; and (j) SSMRN.

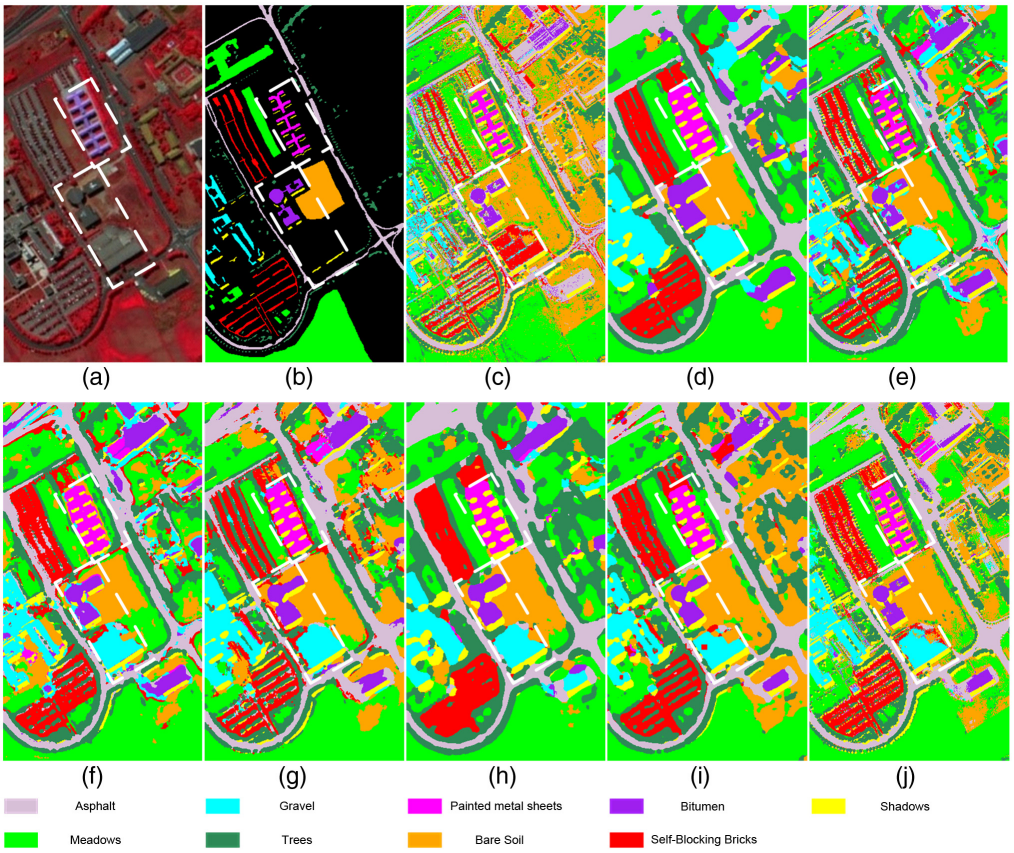

Fig. 7 PU data set and classification maps using different methods: (a) false-color image; (b) ground-truth map; (c) Bi-GRU; (d) MRCNN; (e) SSUN; (f) SSAN; (g) RSSAN; (h) MorphCNN; (i) SSFTT; and (j) SSMRN.

to the ground-truth map. Due to the lack of spatial features, classification maps of Bi-GRU suffer from the pepper noise and misclassification inside an object. Compared with spectral FE methods, spatial FE methods make full use of the continuity of the ground object and yield a cleaner classification map. The main problem of MRCNN lies in the over-smoothing phenomenon. SSRAN, MorphCNN, and SSFTT have the over-smoothing phenomenon, too. They belong to the way of directly extracting joint deep spectral-spatial features from original data or several principal components of the original data, and spectral features come from the patch scale. Meanwhile, SSMRN, SSUN, and SSAN can better retain the detailed boundary of different objects, and acquire more smooth and homogeneous results, especially within the white dashed box. The most likely reason is that they have spatial and spectral FE branches, and spectral features come from the pixel scale. But SSUN and SSAN do not consider the weight relationship between the two branches depending on the spatial resolution of images. The proposed SSMRN takes the weight between spectral and spatial features into consideration and can further reduce over-smoothing.

5 Discussion

The experimental results of the three public data sets indicate that SSMRN has a more competitive performance in terms of three measurements (CA, OA, and Kappa) and classification maps than all the compared methods. This is due to:

1. The SSMRN is designed with a spectral branch and a spatial branch to extract spectral-spatial features. These operations join spectral features and spatial information together sufficiently.

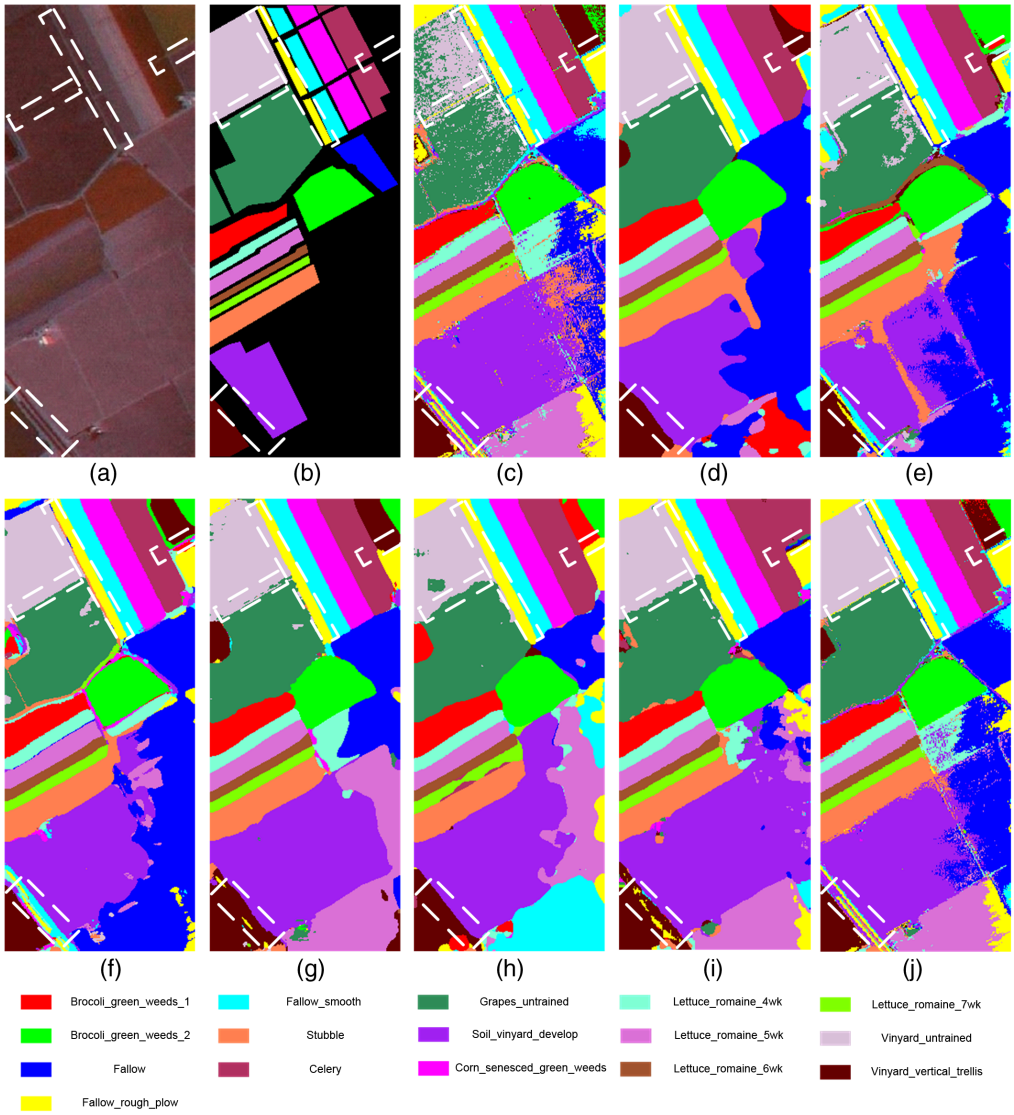

Fig. 8 SA data set and classification maps using different methods on the: (a) false-color image; (b) ground-truth map; (c) Bi-GRU; (d) MRCNN; (e) SSUN; (f) SSAN; (g) RSSAN; (h) MorphCNN; (i) SSFTT; and (j) SSMRN.

2. The proposed framework takes the weight between spectral and spatial features into consideration and can reduce over-smoothing. Meanwhile, the multi-task learning technology is integrated into the framework, improving the stability of results.

6 Conclusion

To significantly reduce the over-smoothing effect and effectively learn the features of objects, a multi-task learning SSMRN has been proposed to extract spectral-spatial features. The experimental results of the three public data sets demonstrate that the method not only mitigates the over-smoothing phenomenon, but also has a better performance compared with the other methods in terms of CA, OA, and Kappa. Our method significantly outperforms other methods under different training sample conditions.

Although we utilize the proposed band Bi-GRU and MRCNN as the spectral and spatial feature extractors in the implementation of the proposed SSMRN, other deep networks can also be introduced into our model, especially for spectral extractors. It deserves to be investigated in future work.

Acknowledgments

This work was supported in part by the Major Science and Technology Program of Henan Province (Grant Nos. 222102320341, 212102311149, and 212102310432), by the key scientific research projects of colleges and universities in Henan Province (Grant No. 22B420004), by the Fundamental Research Funds for the Universities of Henan Province (Grant No. NSFRF210401), Doctoral Foundation of Henan Polytechnic University (Grant No. B2017-09, B2017-14, and B2015-22), by the National Natural Science Foundation of China (Grant No. 41801318). We declare that we do not have any commercial or associative interest that represents a conflict of interest in connection with the work submitted.

References

1. H. Sun et al., "Spectral–spatial attention network for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.* **58**(5), 3232–3245 (2020).
2. Y. Xu et al., "Spectral–spatial unified networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.* **56**(10), 5893–5909 (2018).
3. L. Zhang et al., "Simultaneous spectral-spatial feature selection and extraction for hyperspectral images," *IEEE Trans. Cybern.* **48**(1), 16–28 (2018).
4. M. Ahmad et al., "Hyperspectral image classification—traditional to deep models: a survey for future prospects," *IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens.* **15**, 968–999 (2022).
5. W. Liu et al., "A survey of deep neural network architectures and their applications," *Neurocomputing* **234**, 11–26 (2017).
6. P. Zhou et al., "Learning compact and discriminative stacked autoencoder for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.* **57**(7), 4823–4833 (2019).
7. M. Ahmad et al., "Multi-layer extreme learning machine-based autoencoder for hyperspectral image classification," in *VISIGRAPP (4: VISAPP)*, pp. 75–82 (2019).
8. B. Liu et al., "Spatial–spectral jointed stacked auto-encoder-based deep learning for oil slick extraction from hyperspectral images," *J. Indian Soc. Remote Sens.* **47**(12), 1989–1997 (2019).
9. B. Ayhan and C. Kwan, "Application of deep belief network to land cover classification using hyperspectral images," *Lect. Notes Comput. Sci.* **10261**, 269–276 (2017).
10. G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation* **18**(7), 1527–1554 (2006).
11. S. K. Roy et al., "Morphological convolutional neural networks for hyperspectral image classification," *IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens.* **14**, 8689–8702 (2021).
12. L. Zhang, L. Zhang, and B. Du, "Deep learning for remote sensing data: a technical tutorial on the state of the art," *IEEE Geosci. Remote Sens. Mag.* **4**(2), 22–40 (2016).
13. J. Yang, Y.-Q. Zhao, and J.C.-W. Chan, "Learning and transferring deep joint spectral–spatial features for hyperspectral classification," *IEEE Trans. Geosci. Remote Sens.* **55**(8), 4729–4742 (2017).
14. Z. Zhong et al., "Spectral–spatial residual network for hyperspectral image classification: a 3-D deep learning framework," *IEEE Trans. Geosci. Remote Sens.* **56**(2), 847–858 (2018).
15. M. Zhu et al., "Residual spectral–spatial attention network for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.* **59**(1), 449–462 (2021).
16. Y. Xu, B. Du, and L. Zhang, "Beyond the patchwise classification: spectral-spatial fully convolutional networks for hyperspectral image classification," *IEEE Trans. Big Data* **6**(3), 492–506 (2020).
17. S. Wan et al., "Hyperspectral image classification with context-aware dynamic graph convolutional network," *IEEE Trans. Geosci. Remote Sens.* **59**(1), 597–612 (2021).
18. L. Sun et al., "Spectral–spatial feature tokenization transformer for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.* **60**, 1–14 (2022).

19. R. Hang et al., "Cascaded recurrent neural networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.* **57**(8), 5384–5394 (2019).
20. F. Zhou et al., "Hyperspectral image classification using spectral-spatial LSTMs," *Neurocomputing* **328**, 39–47 (2019).
21. X. Mei et al., "Spectral-spatial attention networks for hyperspectral image classification," *Remote Sens.* **11**(8), 963 (2019).
22. M. Seydgar et al., "3-D convolution-recurrent networks for spectral-spatial classification of hyperspectral images," *Remote Sens.* **11**(7), 883 (2019).
23. M. V. Valueva et al., "Application of the residue number system to reduce hardware costs of the convolutional neural network implementation," *Math. Comput. Simul.* **177**, 232–243 (2020).
24. C. Zhang et al., "A study on overfitting in deep reinforcement learning," arXiv:1804.06893 (2018).
25. Y. Xu, B. Du, and L. Zhang, "Assessing the threat of adversarial examples on deep neural networks for remote sensing scene classification: attacks and defenses," *IEEE Trans. Geosci. Remote Sens.* **59**(2), 1604–1617 (2021).
26. C. Cheng et al., "Hyperspectral image classification via spectral-spatial random patches network," *IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens.* **14**, 4753–4764 (2021).
27. Y. Xu et al., "Hyperspectral image classification via a random patches network," *ISPRS J. Photogramm. Remote Sens.* **142**, 344–357 (2018).
28. H.-C. Shin et al., "Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning," *IEEE Trans. Med. Imaging* **35**(5), 1285–1298 (2016).
29. L. Ma et al., "Deep learning in remote sensing applications: a meta-analysis and review," *ISPRS J. Photogramm. Remote Sens.* **152**, 166–177 (2019).
30. K. He et al., "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. and Pattern Recognit.*, Las Vegas, Nevada, pp. 770–778 (2016).
31. L. Mou, P. Ghamisi, and X. X. Zhu, "Deep recurrent neural networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.* **55**(7), 3639–3655 (2017).
32. S. Li et al., "Deep learning for hyperspectral image classification: an overview," *IEEE Trans. Geosci. Remote Sens.* **57**(9), 6690–6709 (2019).

Shi He is an assistant professor at the Henan Polytechnic University. He received his BS degree from China Agricultural University in 2009, respectively, and his PhD in cartography and geographic information system from Beijing Normal University in 2016. His current research interests include optical remote sensing, machine learning, and image classification.

Huazhu Xue received his PhD in cartography and geographic information systems in the area of quantitative remote sensing from the School of Geography, Beijing Normal University, Beijing, China, in 2012. Since 2012, he has been an associate professor at the School of Surveying and Land Information Engineering, Henan Polytechnic University, Jiaozuo, China. His research interests include vegetation parameters inversion, satellite image processing, and GIS applications.

Jiehai Cheng is an associate professor at Henan Polytechnic University. He received his PhD in cartography and geographic information system from Beijing Normal University in 2013. His current research interests include high-resolution remote sensing, deep learning, image classification, and GIS intelligent analysis.

Lei Wang received his PhD in cartography and geographical information engineering from China University of Mining and Technology, Beijing, in 2016. Since 2016, he has been an associate professor at Henan Polytechnic University in Jiaozuo City. His current research interests include global GIS modeling, discrete global grid, and the application of remote sensing.

Yaping Wang is an associate professor at the Henan Polytechnic University. She received her PhD in cartography and geographic information engineering from China University of Mining

and Technology, Beijing, in 2014. Her current research interests include image classification, remote sensing of water resources, and GIS applications.

Yongjuan Zhang received her bachelor's degree from Henan Polytechnic University in June 2022. Currently, she is studying for a master's degree in Nanjing Normal University. Her research direction is target extraction based on deep learning.