Spectral–Spatial Feature Tokenization Transformer for Hyperspectral Image Classification

Le Sun, *Member, IEEE*, Guangrui Zhao, Yuhui Zheng, *Member, IEEE*, and Zebin Wu, *Senior Member, IEEE*

***Abstract*— In hyperspectral image (HSI) classification, each pixel sample is assigned to a land-cover category. In the recent past, convolutional neural network (CNN)-based HSI classification methods have greatly improved performance due to their superior ability to represent features. However, these methods have limited ability to obtain deep semantic features, and as the layer's number increases, computational costs rise significantly. The transformer framework can represent high-level semantic features well. In this article, a spectral–spatial feature tokenization transformer (SSFTT) method is proposed to capture spectral–spatial features and high-level semantic features. First, a spectral–spatial feature extraction module is built to extract low-level features. This module is composed of a 3-D convolution layer and a 2-D convolution layer, which are used to extract the shallow spectral and spatial features. Second, a Gaussian weighted feature tokenizer is introduced for features transformation. Third, the transformed features are input into the transformer encoder module for feature representation and learning. Finally, a linear layer is used to identify the first learnable token to obtain the sample label. Using three standard datasets, experimental analysis confirms that the computation time is less than other deep learning methods and the performance of the classification outperforms several current state-of-the-art methods. The code of this work is available at https://github.com/zgr6010/HSI_SSFTT for the sake of reproducibility.**

***Index Terms*— Convolutional neural networks (CNNs), hyperspectral image (HSI) classification, semantic features, spectral–spatial tokenization, transformer.**

I. Introduction

IN RECENT years, the information captured by hyperspectral sensors has become more accurate information with the increasing spatial and spectral resolutions due to advances in spectral imaging technology [1]. The resulting hyperspectral image (HSI) comprises 2-D spatial information and 1-D spectral information about the objects. With its abundant information, an HSI can be used for various purposes, such as precision agriculture [2], biomedical imaging [3], mineral exploration [4], food safety [5], and military reconnaissance [6]. Several data processing techniques have been explored to harness the full potential of the HSI data, such as denoising [7], [8], unmixing [9], target detection [10], and classification [11]–[13]. Among these HSI processing techniques, land-cover information classification has attracted much attention.

Numerous traditional HSI classification methods have been proposed, such as the k-nearest neighbor [14], the Bayesian estimation method [15], the multinomial logistic regression [16], [17], and the support vector machine (SVM) [18]–[21]. In addition, several methods for dimension reduction and spectral information extraction have also been developed, including the principal component analysis (PCA) [22], [23], the independent component analysis (ICA) [24], and the linear discriminant analysis (LDA) [25]–[27]. However, these methods ignore the spatial correlation among the pixels in spatial dimension and fail to fully use the spatial features. For the optimal extraction of image spatial features, a variety of mathematical morphological operators have been developed, such as morphological profile (MP) [28], extended MP (EMP) [29], and extended multiattribute profile (EMAP) [30]. More specifically, to effectively utilize the correlation among the pixels, Sun *et al.* [13] developed an adjacent superpixel-based multiscale spatial–spectral kernel method to improve classification results. The HSI data are fully exploited in terms of its spatial and spectral content in the method. Duan *et al.* [31] developed a semisupervised geodesic-based sparse manifold hypergraph method to improve the classification performance. This method organically combined the hypergraph embedded feature extraction and sparse representation to obtain the nonlinear discriminative features of HSI. Moreover, in [32], a multistructure unified discriminant embedding method was proposed to obtain more discriminative features and further improve the classification accuracies. Yet, none of these methods uses a deep model.

Advances in rapid deep learning technology development have accelerated the development of image processing techniques in a variety of fields, also contributing to the technical innovation of remote sensing image processing [33]. For the HSI, numerous different classification methods using deep

Manuscript received September 11, 2021; revised November 25, 2021 and December 31, 2021; accepted January 14, 2022. Date of publication January 18, 2022; date of current version March 14, 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 61971233, Grant 62076137, Grant U20B2065, and Grant U20B2061; in part by the Natural Science Foundation of Jiangsu Province under Grant BK 20211539; in part by the Henan Key Laboratory of Food Safety Data Intelligence under Grant KF2020ZD01; and in part by the Postgraduate Research & Practice Innovation Program of Jiangsu Province under Grant KYCX21_1004. *(Corresponding author: Yuhui Zheng.)*

Le Sun is with the School of Computer and Science, Nanjing University of Information Science and Technology, Nanjing 210044, China, and also with the Engineering Research Center of Digital Forensics, Ministry of Education, Nanjing University of Information Science and Technology (NUIST), Nanjing 210044, China (e-mail: sunlecncom@163.com).

Guangrui Zhao and Yuhui Zheng are with the School of Computer and Science, Nanjing University of Information Science and Technology, Nanjing 210044, China (e-mail: cs_zhaogr@nuist.edu.cn; zhengyh@vip.126.com).

Zebin Wu is with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: zebin.wu@gmail.com).

Digital Object Identifier 10.1109/TGRS.2022.3144158

models have been proposed. Chen *et al.* [34], [35] applied stacked autoencoders (SAEs) and a deep belief network (DBN) for spatial–spectral features' hierarchical extraction. These methods require that the image patches of the training samples be flattened into 1-D features as input. However, in doing so, the original image is altered in terms of spatial information. Subsequently, with the popularity of the convolutional neural network (CNN), several CNN-based network structures were investigated for HSI classification. An improved classification process was proposed by Hu *et al.* [36] by using a 1-D-CNN with five convolutional layers. This method takes the spectral information carried by HSI as input and effectively extracts the spectral features. In [37] and [38], a 2-D-CNN model was proposed to extract the spatial features carried by the first few principal components after dimension reduction. In [39], to extract spatial and spectral features, a dual-branch network structure was developed by combining 2-D- and 1-D-CNN technologies. In the end, these features were combined into a fully connected layer so that joint spectral–spatial features could be extracted for classification. To better extract spatial–spectral features, Chen *et al.* [40] introduced a new 3-D-CNN method for the effective extraction of 3-D features. Roy *et al.* [41] combined the characteristics of 3-D- and 2-D-CNNs and proposed a hierarchical network structure. In addition to fully extracting spatial–spectral features, this method could reduce the computational complexity and improve classification accuracy.

With the prevalence of residual networks in the field of image classification, Zhong *et al.* [42] proposed a spatial–spectral residual network to classify HSI. This method greatly improved the utilization rate of features, taking the information of the front layer features as the supplement of the rear layer features. In [43], a pyramid residual network was developed to classify HSI. This model increased the dimension of the feature map and grouped the features into pyramid residual blocks, effectively extracting the feature information uncovered by the convolution filter. In [44], an end-to-end dense convolutional network framework was proposed for the extraction of spatial–spectral features. Furthermore, Wang *et al.* [45] improved the dense convolutional network structure and proposed a new Cubic-CNN framework for feature extraction. This method took the extracted original image patches and the features after the dimension reduction and 1-D convolutional operation as the inputs to the network. This process effectively reduced the features' redundancy. In [46], a new supervised multiscale alternately updated clique network was proposed for HSI classification to fully exploit features in different scales. In this method, a multiscale alternately updated clique block was designed, which applied convolution kernels of various sizes to adaptively utilize multiscale information. Most of the above methods are based on the CNN backbone and its variants. Although these methods effectively improve the HSI classification performance, the reduced classification performance caused by the limited training samples and the increase of network layers is difficult to overcome. They also have excessive feature redundancy.

In addition to CNN, some other categories of networks with good performance are also used to classify HSI. As a widely applied model in the field of image segmentation, a fully convolutional network (FCN) [47], [48] has been successfully employed in HSI classification tasks. Using convolution and pooling layers, this model can effectively learn the deep features of HSI and improve the classification performance. Moreover, as a feedforward neural network, the recurrent neural network (RNN) [49] was studied for the HSI classification task. An RNN can construct a sequence model to effectively simulate the relationship between adjacent spectral bands. Wu and Prasad [50] effectively combined the CNN and RNN and proposed a convolutional RNN structure. First, several convolution layers were used to extract the former convolution features. Then, the recurrent layers were used to further extract the contextual information from the convolution features. In the meantime, other networks based on generative adversarial networks (GANs) [51], [52], capsule networks [53], and graph convolutional networks (GCNs) [54] have also been introduced for HSI classification. These backbone networks model the relationship between pixels well and exhibit robustness.

Recently, a new model called vision transformer (ViT) [55] performed favorably in the image processing field. Some work has been done to apply the transformer models to HSI classification. In [56], a spatial–spectral transformer (SST) model was proposed. The author extracted spatial features by using the network structure similar to VGGNet [57] and constructed the relationship between adjacent spectra with dense transformers. Finally, the classification results were obtained by using the multilayer perceptron (MLP). Similarly, Qing *et al.* [58] effectively captured the continuous spectral relationship by introducing a spectral attention mechanism, which was combined with the multiattention mechanism in the transformer. More recently, Hong *et al.* [59] developed a new model called SpectralFormer (SF), which can learn spectral representation information from groupwise neighboring bands and construct a cross-layer transformer encoder (TE) module. However, the abovementioned methods are all improved transformer methods based on spectral information processing. Although the transformer performs outstandingly in capturing spectral signatures, it loses the power in capturing local semantic features and makes insufficient use of image spatial information.

The original transformer [60] is a model applied in natural language processing (NLP) based on the self-attention (SA) mechanism. The input to the model is a sequence of tokens. Multihead attention is used to draw global correlations in the input token sequence. Therefore, to take advantage of the transformer's ability to obtain local spatial semantic information and model the relationship between adjacent sequences, a spectral–spatial feature tokenization transformer (SSFTT) model for HSI classification is proposed. First, in this model, a 3-D convolution layer and a 2-D convolution layer are used to extract the shallow spectral–spatial features. This effectively reduces the feature redundancy and inaccuracies caused by the increased number of layers. Second, the flattened features are tokenized by the Gaussian weighed tokenizer. Then, the

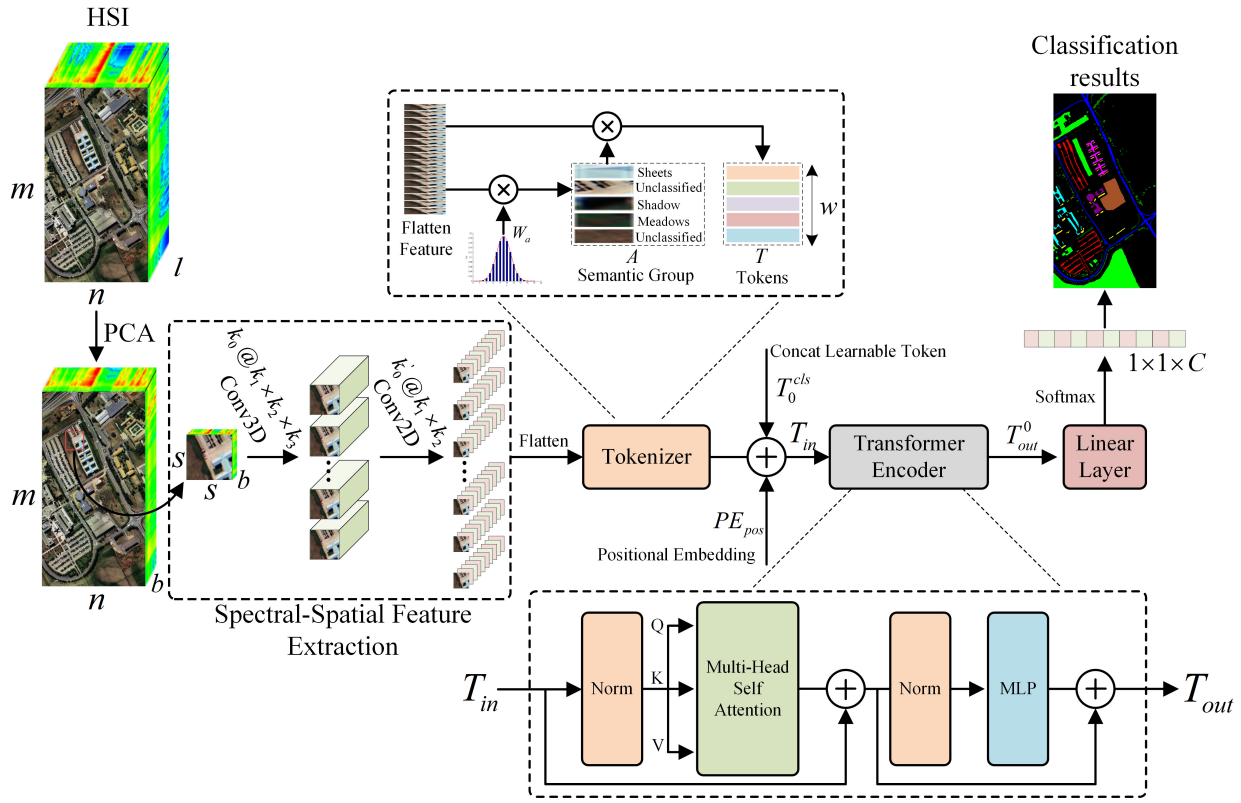

Fig. 1. Overall framework of the proposed SSFTT network for the HSI classification.

generated tokens are used as input to the TE module. Finally, a softmax-based linear classifier is adopted to determine the label of each pixel.

The main contributions of this article are summarized as follows.

1) A simple and efficient hierarchical CNN module is proposed in our SSTFF network for extracting shallow spatial–spectral features. It only consists of one 3-D convolution layer and one 2-D convolution layer. Then, this module is combined with the transformer structure to develop a new lightweight network to replace a single CNN structure for alleviating the computational cost.
2) A Gaussian distribution weighted tokenization module is proposed to transform the shallow spatial–spectral features to the tokenized semantic features. Its function is to make the deep semantic features expressed by the tokens more in line with the distribution characteristics of the samples, thereby making the samples more separable.
3) The systematic combination of the CNN network and the transformer structure from shallow to deep can fully exploit the spectral–spatial information in the HSI and express the low-middle-deep semantic features of the HSI concisely and efficiently, thereby significantly improving the classification accuracy. Experiments on three representative datasets verify the superiority of the proposed network.

The remainder of this article is organized as follows. The proposed SSFTT model is introduced in Section II. Section III shows the experimental datasets, the design of experimental parameters, and the comparison of classification accuracies. Some related conclusions are presented in Section IV.

II. Materials and Methods

Fig. 1 shows the overall framework for the HSI classification based on the proposed SSFTT model composed of three parts—the spectral–spatial feature extraction, the Gaussian-weighted feature tokenizer, and the TE module.

A. Spectral–Spatial Feature Extraction

Original HSI data $\boldsymbol{I} \in \mathbb{R}^{m\times n\times l}$ are given, where $m \times n$ is the spatial size and l is the number of spectral bands. Each pixel in $\boldsymbol{I}$ has l spectral dimension and forms a one-hot category vector $Y = (y_1, y_2, \ldots, y_C) \in \mathbb{R}^{1\times 1\times C}$, where C is the number of land-cover classes. Thus, HSI is composed of l bands that carry useful spectral information but also result in large dimensions, adding significant computation. Therefore, PCA is applied to process the HSI data in order to reduce the amount of computation and spectral dimension. PCA reduces the number of bands from l to b and keeps the spatial dimension unchanged. Hence, the HSI data after PCA dimension reduction are expressed as $\boldsymbol{I}_{\text{pca}} \in \mathbb{R}^{m\times n\times b}$, where b is the number of spectral bands after PCA.

Next, 3-D-patch extraction is performed on the HSI data $\boldsymbol{I}_{\text{pca}}$. Each 3-D adjacent patch $\boldsymbol{P} \in \mathbb{R}^{s\times s\times b}$ is created from $\boldsymbol{I}_{\text{pca}}$, where $s \times s$ indicates the window size. The center pixel position of each patch is set to (x_i, x_j), where $0 \le i < m$, $0 \le j < n$. The true label of each patch is determined by

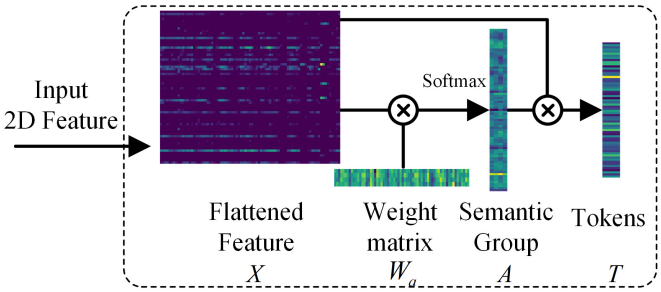

Fig. 2. Visualization process of tokenizer.

the center pixel's label. When the patch around a single pixel is extracted, the edge pixels cannot be retrieved. Therefore, a padding operation is performed for these pixels. The width of the padding is $(s-1)/2$. Thus, the final number of 3-D-patches generated from $\boldsymbol{I}_{\text{pca}}$ is given by $m \times n$. Each patch covers the width from $x_i - (s-1)/2$ to $x_i + (s-1)/2$, height from $x_j - (s-1)/2$ to $x_j + (s-1)/2$, and all b spectral bands. After the pixel patches with zero labels are removed, all the remaining sample patches are divided into a training sample patch set and a test sample patch set.

Then, two convolution layers (3-D and 2-D) are used to extract spectral–spatial features of each sample patch. Each training sample patch of size $s \times s \times b$ is used as the input data to the 3-D convolution layer. In the 3-D convolution layer, the calculated value at a spatial position (α, β, γ) for the jth feature cube of the ith layer is given by

$$v_{i,j}^{\alpha,\beta,\gamma} = \Phi\left(\sum_{k}\sum_{h=0}^{H_i-1}\sum_{w=0}^{W_i-1}\sum_{r=0}^{R_i-1}\omega_{i,j,k}^{h',w',r'}\, v_{i-1,k}^{\alpha+h',\beta+w',\gamma+r'} + b_{i,j}\right) \tag{1}$$

where $\Phi(\cdot)$ is the activation function and k is the feature cube related to the jth feature cube in the $(i-1)$th layer. H_i, W_i, and R_i, respectively, represent the width, height, and channel number of the 3-D convolution kernel. In this case, R_i stands for spectral dimension. $\omega_{i,j,k}^{h',w',r'}$ is the weight parameter of position (h', w', r') connected to the kth feature cube, and $b_{i,j}$ is the bias.

In this model, the 3-D convolution layer is composed of k_0 3-D kernels theoretically. The size of each 3-D kernel is $k_1 \times k_2 \times k_3$. By 3-D convolution, k_0 3-D feature cubes covering spectral–spatial information are generated. The size of each cube is $(s-k_1+1) \times (s-k_2+1) \times (b-k_3+1)$. The total size of the feature cube is $k_0@(s-k_1+1) \times (s-k_2+1) \times (b-k_3+1)$.

After the rearrangement operation, the input size as the feature of the next 2-D convolution layer is $(s-k_1+1) \times (s-k_2+1) \times k_0(b-k_3+1)$. In the 2-D convolution layer, the activated value $v_{i,j}^{\alpha,\beta}$ at a spatial position (α, β) on the jth feature map in the ith layer is defined as

$$v_{i,j}^{\alpha,\beta} = \Phi\left(\sum_{k}\sum_{h=0}^{H_i'-1}\sum_{w=0}^{W_i'-1}\omega_{i,j,k}^{h',w'}\, v_{i-1,k}^{\alpha+h',\beta+w'} + b_{i,j}\right) \tag{2}$$

where H_i' and W_i' represent the width and height of the 2-D convolution kernel, respectively. $\omega_{i,j,k}^{h',w'}$ is the weight parameter of position (h', w') connected to the kth feature map.

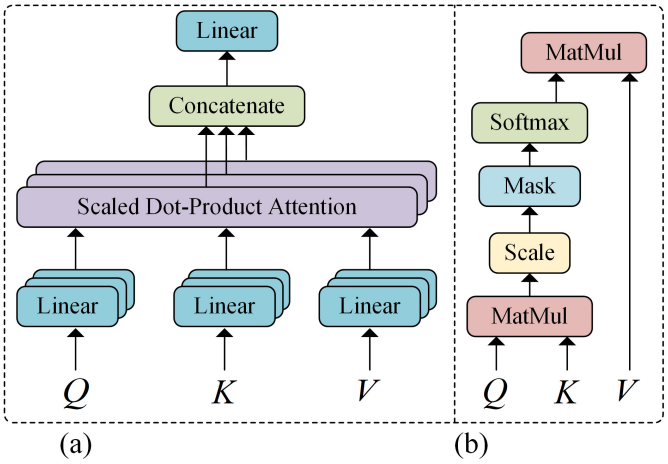

Fig. 3. Illustration of the attention mechanism in the TE module. (a) MSA module. (b) SA.

In this model, the total size of the feature maps generated by 2-D convolution is $k_0'@[s-2\times(k_1+1)]\times[s-2\times(k_2+1)]$, where k_0' is the number of 2-D convolution kernels. The size of each 2-D kernel is $k_1 \times k_2$.

B. Gaussian-Weighted Feature Tokenizer

The features extracted by two layers of convolution operation carry spectral and spatial information, but they cannot adequately describe the features of ground objects. Therefore, feature maps are further defined as semantic tokens, which can represent and process the high-level semantic concepts of HSI feature categories. For this part, the input flattened feature map is defined as $\boldsymbol{X} \in \mathbb{R}^{uv\times z}$, where u is height, v is width, and z is the number of channels. Feature tokens are defined as $\boldsymbol{T} \in \mathbb{R}^{w\times z}$, where w represents the number of tokens.

For feature map $\boldsymbol{X}$, $\boldsymbol{T}$ can be obtained by the following formula:

$$\boldsymbol{T} = \underbrace{\text{softmax}(\boldsymbol{X}\boldsymbol{W}_a)^T}_{\boldsymbol{A}}\boldsymbol{X}. \tag{3}$$

Here, $\boldsymbol{W}_a \in \mathbb{R}^{z\times w}$ represents a weight matrix initialized with a Gaussian distribution, and $\boldsymbol{X}\boldsymbol{W}_a$ represents that they perform the 1×1 pointwise product. The goal is to map $\boldsymbol{X}$ into the semantic group. The size of the semantic group obtained through this step is $\boldsymbol{A} \in \mathbb{R}^{uv\times w}$. Then, $\boldsymbol{A}$ is transposed, and softmax$(\cdot)$ is used to focus on the relatively important semantic part. Finally, $\boldsymbol{A}$ multiplies with $\boldsymbol{X}$ to make $\boldsymbol{T}$ semantic tokens. To visualize the actual form of the tokenizer, Fig. 2 illustrates a sample of the transformation process.

C. Transformer Encoder Module

As shown in Fig. 1, the semantic tokens generated in Section II-B serve as inputs into the TE module to learn the relationships between high-level semantic features. This module consists mainly of three subparts.

As the first subpart, the position information of each semantic token is marked using position embedding. Each token is

represented by $[T_1, T_2, \ldots, T_w]$. The tokens are concatenated with a learnable classification token T_0^{cls}, which is used to perform the classification task. Then, the positional information PE_{pos} is encoded and appended into the token representations. The resulting embedded sequence of semantic tokens is given by

$$\boldsymbol{T}_{\text{in}} = \left[T_0^{\text{cls}}, T_1, \ldots, T_w\right] + \text{PE}_{\text{pos}}. \tag{4}$$

The second and essential subpart is the TE. This block is designed to model deep relationships among the semantic tokens. It contains a multihead SA (MSA) block [see Fig. 3(a)], an MLP layer, and two normalization layers (LN). Residual skip connections are designed before the MSA block and MLP layer.

The transformer structure behaves well because of its core MSA block. The use of an SA mechanism [see Fig. 3(b)] in this block effectively captures the correlation among feature sequences. In order to learn multiple meanings, three learnable weight matrices, $\boldsymbol{W}_Q$, $\boldsymbol{W}_K$, and $\boldsymbol{W}_V$, are defined in advance, and the tokens are linearly mapped to form 3-D-invariant matrices, including queries $\boldsymbol{Q}$, keys $\boldsymbol{K}$, and values $\boldsymbol{V}$ by three learnable weight matrices. The attention score is calculated using all of $\boldsymbol{Q}$ and $\boldsymbol{K}$, and the weight of the score is calculated using the softmax function. In summary, SA is formulated as follows:

$$\text{SA} = \text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_K}}\right)\boldsymbol{V} \tag{5}$$

where d_K is the dimension of $\boldsymbol{K}$.

The MSA block involves multiple groups of the weight matrix in mapping $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$, using the same operation process to calculate the multihead attention value. Then, every head attention results are concatenated together. This process is expressed by this equation

$$\text{MSA}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{Concat}(\text{SA}_1, SA_2, \ldots, \text{SA}_h)\boldsymbol{W} \tag{6}$$

where h is the head number, $\boldsymbol{W}$ is the parameter matrix, and $\boldsymbol{W} \in \mathbb{R}^{h \times d_K \times d_w}$, where $d_w = w$ (number of tokens).

Next, the weight matrix learned in the previous step is entered into the MLP layer. The MLP is made up of two fully connected layers. Between this pair has a nonlinear activation function named the Gaussian error linear unit. The MLP layer is followed by an LN, which improves gradient exploding, reduces vanishing gradient problems, and enables faster training.

Through the TE module, the sizes of the input T_{in} and output T_{out} are equal. The classification token vector T_{out}^0 is the input to the linear layer at the top for the final classification. Through the linear layer, the probability that the input belongs to a certain category is calculated by the softmax function. The label with the largest probability value is the category of the sample.

D. Implementation

Compared with the backbone CNN, the SSFTT reduces the number of network layers. In addition, it can model at the semantic level of the image patch by introducing the tokenizer and TE. Here, the Pavia University dataset of size $610 \times 340 \times 103$ is selected as an example to illustrate the designed SSFTT model.

Algorithm 1 SSFTT Model

Input: Input an HSI data $\boldsymbol{I} \in \mathbb{R}^{m \times n \times l}$ and ground-truth $\boldsymbol{Y} \in \mathbb{R}^{m \times n}$; PCA bands number $b = 30$; patch size $s = 13$; training sample rate $\mu\%$.

Output: Predicted labels of the test dataset.

Set batchsize to 64, optimizer Adam (learning rate: $1e-3$), epoches number ϵ to 100.
Obtain the $\boldsymbol{I}_{\text{pca}}$ after PCA tranform.
Create all sample patches in the $\boldsymbol{I}_{\text{pca}}$, and divide them into training dataset and test dataset.
Generate training loader and test loader.
for $i = 1$ to ϵ **do**
    Perform 3D convolution layer and 2D convolution layer.
    Flatten each 2D feature map into 1D feature vector.
    Perform tokenization tranform with feature vector and the initialized weight to generate semantic tokens.
    Concatenate the learnable tokens to form the semantic tokens and embed position on the semantic tokens.
    Perform TE module.
    Input the first classification token to the last linear layer.
    Use the softmax function to identify the labels.
end for
Use test dataset with the trained model to get predicted labels.

After PCA dimensional reduction and patch extraction, the size of each patch is $13 \times 13 \times 30$. In the first 3-D convolution layer, eight $11 \times 11 \times 28$ feature cubes are generated by convolution operation with eight $3 \times 3 \times 3$ cube kernels on each patch. 3-D convolution is used in this step because abundant spectral information is stored in each patch. The eight feature cubes are rearranged to generate one $11 \times 11 \times 224$ feature cube. Then, 2-D convolution with 64 3×3 plane kernels is used to obtain 64 feature maps, each sized 9×9. Each feature map is flattened into a 1-D feature vector to obtain 64 vectors of size 1×81. At this point, the feature obtained is equivalent to $\boldsymbol{X} \in \mathbb{R}^{81 \times 64}$ in this article.

In the next step, the Xavier standard normal distribution is used to get the initial weight matrix $\boldsymbol{W}_a \in \mathbb{R}^{64 \times 4}$ to guide the feature distribution to be more regular. The initialized weight matrix $\boldsymbol{W}_a \in \mathbb{R}^{64 \times 4}$ is multiplied by the feature vector group to make the semantic group $\boldsymbol{A} \in \mathbb{R}^{81 \times 4}$. Then, the transpose of $\boldsymbol{A}$ is multiplied by $\boldsymbol{X}$ to get the final semantic tokens $\boldsymbol{T} \in \mathbb{R}^{4 \times 64}$. An all-zero vector is concatenated into $\boldsymbol{T}$ as a learnable token and embedded with learned position markers to obtain $\boldsymbol{T}_{\text{in}} \in \mathbb{R}^{5 \times 64}$. Processing $\boldsymbol{T}_{\text{in}}$ through the TE module, semantic features are represented. This module has the same size of input and output. The output of the first classification token $T_{\text{out}}^0 \in \mathbb{R}^{1 \times 64}$ is taken out as a classification vector. This vector is input into the softmax-based linear classifier to obtain the judged label. The overall process of the proposed SSFTT method is shown in Algorithm 1.

TABLE I

TRAINING AND TEST SAMPLE NUMBERS IN THE INDIAN PINES DATASET, THE PAVIA UNIVERSITY DATASET, AND THE HOUSTON 2013 DATASET

NO.	Indian Pines			Pavia University			Houston 2013		
	Class	Training.	Test.	Class	Training.	Test.	Class	Training.	Test.
1	Alfalfa	5	41	Asphalt	332	6299	Healthy Grass	125	1126
2	Corn-notill	143	1285	Meadows	932	17717	Stressed Grass	125	1129
3	Corn-mintill	83	747	Gravel	105	1994	Synthetis Grass	70	627
4	Corn	24	213	Trees	153	2911	Tree	124	1120
5	Grass-pasture	48	435	Metal Sheets	67	1278	Soil	124	1118
6	Grass-tree	73	657	Bare soil	251	4778	Water	33	292
7	Grass-pasture-mowed	3	25	Bitumen	67	1263	Residential	127	1141
8	Hay-windrowed	48	430	Bricks	184	3498	Commercial	124	1120
9	Oats	2	18	Shadows	47	900	Road	125	1127
10	Soybean-notill	97	875				Highway	123	1104
11	Soybean-mintill	245	2210				Railway	123	1112
12	Soybean-clean	59	534				Parking Lot 1	123	1110
13	wheat	20	185				Parking Lot 2	47	422
14	woods	126	1139				Tennis Court	43	385
15	Buildings-Grass-Trees	39	347				Running Track	66	594
16	Stone-Steel-Towers	9	84						
-	Total	1024	9225	Total	2138	40638	Total	1502	13527

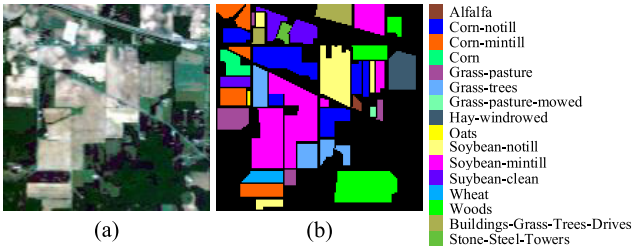

Fig. 4. Indian Pines dataset. (a) False-color map. (b) Ground-truth map.

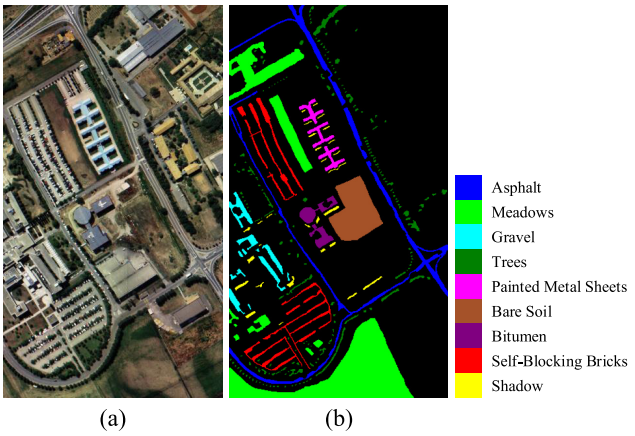

Fig. 5. Pavia University dataset. (a) False-color map. (b) Ground-truth map.

III. EXPERIMENT AND ANALYSIS

A. Data Description

To verify the performance of the proposed method, three classical HSI datasets were selected for experiments, including the Indian Pines, Pavia University, and Houston 2013 datasets.

The Indian Pines dataset was acquired by the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) Sensor over northwestern Indiana, United States (US), in 1992. The uncorrected dataset includes 224 spectral bands ranging from 0.4 to 2.5 μm. It consists of 145×145 pixels with a spatial resolution of 20 m and contains 16 land-cover classes. In the experiment, 24 water-absorption bands and noise bands were removed, and 200 bands were selected. The false-color and ground-truth maps are shown in Fig. 4(a) and (b), respectively.

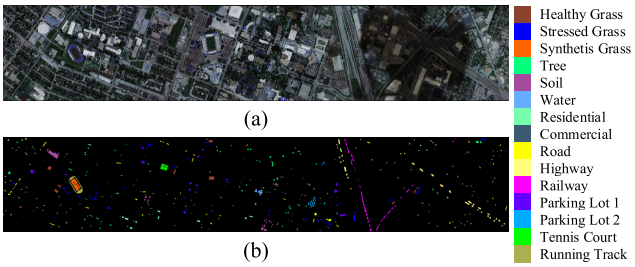

Fig. 6. Houston 2013 dataset. (a) False-color map. (b) Ground-truth map.

The Pavia University dataset was collected by the Reflective Optics System Imaging Spectrometer (ROSIS) Sensor over the Pavia University in Northern Italy in 2001. The uncorrected dataset includes 115 spectral bands ranging from 0.43 to 0.86 μm. The size of this image is 610×340 pixels with a spatial resolution of 1.3 m. Nine land-cover categories are covered. In the experiment, 12 noise bands were removed, and 103 bands were used. Fig. 5(a) and (b) shows the false-color and ground-truth maps, respectively.

The Houston 2013 dataset was provided by the Hyperspectral Image Analysis Group and the NSF-funded Airborne Laser Mapping Center (NCALM) at the University of Houston, US.[1] The dataset was originally used for scientific purposes in the 2013 IEEE GRSS Data Fusion Competition. It comprises 144 spectral bands ranging from 0.38 to 1.05 μm. This dataset is made up of 349×1905 pixels with a spatial resolution of 2.5 m, totaling 15 classes. Fig. 6(a) and (b) shows the false-color and ground-truth maps, respectively.

Table I lists the land-cover categories names, the number of training samples, and test samples regarding these three datasets. Each dataset was divided into a training set and a test set. The Indian Pines and Houston 2013 datasets had a

[1]2013 IEEE GRSS Data Fusion Contest: http://www.grss-ieee.org/community/technical-committees/data-fusion

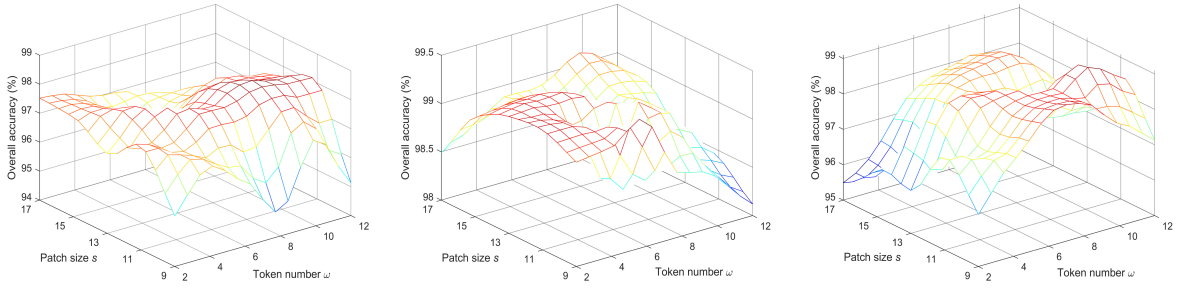

Fig. 7. Impact between different numbers of tokens and patch size for the OA. (a) Indian Pines dataset. (b) Pavia University dataset. (c) Houston 2013 dataset.

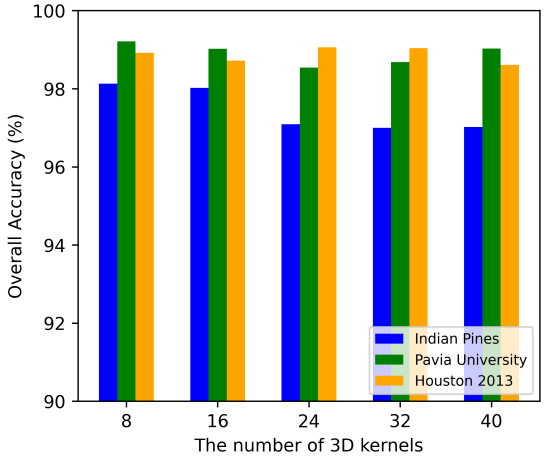

Fig. 8. Impact of 3-D convolution kernels' number on the OA.

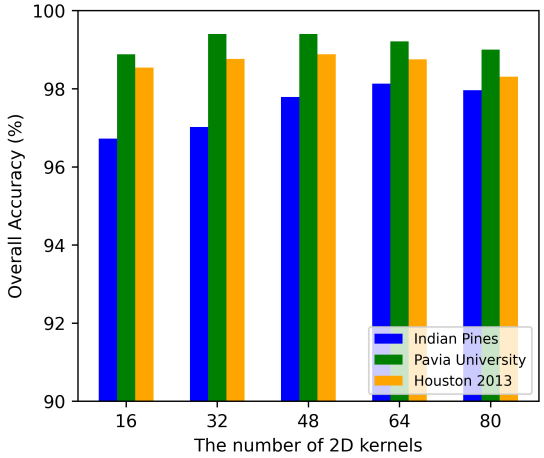

Fig. 9. Impact of 2-D convolution kernels' number on the OA.

TABLE II
IMPACT OF DIFFERENT PATCH SIZES FOR THE OA ON THREE DATASETS (OPTIMAL RESULTS ARE BOLDED)

Patch Size	Indian Pines	Pavia University	Houston 2013
9×9	97.10	99.13	**99.31**
11×11	97.13	99.18	98.71
13×13	**97.47**	**99.21**	98.92
15×15	97.40	99.08	98.20
17×17	97.28	98.70	97.46
19×19	96.96	97.92	96.69

TABLE III
ABLATION ANALYSIS OF THE PROPOSED MODEL ON THE PAVIA UNIVERSITY DATASET (OPTIMAL RESULTS ARE BOLDED)

Cases	Component				Indicators		
	3D Conv	2D Conv	Tokenizer	TE	OA (%)	AA (%)	$\kappa \times 100$
1	√	×	√	√	85.54	69.83	80.58
2	×	√	√	√	94.03	90.60	93.33
3	×	×	PE	√	90.98	85.55	88.06
4	√	√	PE	√	97.51	94.55	95.51
5	√	√	×	×	93.52	92.19	91.33
6	√	√	√	√	**99.21**	**98.69**	**99.15**

random 10% of the total sample number in the training set. 5% of the total sample was randomly selected for the training set from the Pavia University dataset.

B. Experimental Setting

1) Evaluation Indicators: To quantitatively analyze the effectiveness of the proposed method and other methods for comparison, four quantitative evaluation indexes are introduced, including overall accuracy (OA), average accuracy (AA), kappa coefficient (κ), and the classification accuracy of each land-cover category. The larger value of each indicator represents a better classification effect.

2) Configuration: The verification experiments of the proposed method were all implemented in the PyTorch environment, using an Intel Xeon Silver 4210 CPU, 128-GB RAM, and an NVIDIA Geforce RTX 2080Ti 11-GB GPU server. The Adam optimizer was selected as the initial optimizer, and the original learning rate was set to $1e-3$. For batch training, the size of each minibatch was set to 64. Each dataset had 100 training epochs applied.

3) Parameter Analysis: In the parameter analysis, we analyzed several parameters that affect the classification performance and training process, including the patch size $s \times s$ of input cubes, the number of tokens ω, and the number of kernels of two convolution layers. During parameter testing, other parameters, such as batch size, optimizer, and epochs, were configured, as described in Section III-B2.

TABLE IV
CLASSIFICATION PERFORMANCE OBTAINED BY DIFFERENT METHODS FOR THE INDIAN PINES DATASET (OPTIMAL RESULTS ARE BOLDED)

NO.	SVM	EMAP [30]	1D-CNN [36]	2D-CNN [37]	3D-CNN [40]	SSRN [42]	Cubic-CNN [45]	HybridSN [41]	SSFTT
1	65.63	62.50	43.75	48.78	41.46	83.15	87.86	87.80	**95.12**
2	63.44	81.57	77.93	78.13	90.51	95.31	96.35	94.39	**97.67**
3	60.25	83.19	56.72	83.51	79.36	94.23	93.65	96.52	**98.87**
4	41.11	85.89	45.18	47.42	46.01	90.68	82.54	83.89	**91.55**
5	87.05	78.61	87.57	75.12	95.17	97.79	96.69	**98.16**	96.32
6	97.21	79.08	98.63	92.99	**99.70**	98.67	96.69	99.54	99.54
7	89.47	52.63	65.11	60.00	88.00	97.92	90.16	92.97	**100.00**
8	96.66	91.19	97.36	98.37	**100.00**	99.26	98.46	**100.00**	**100.00**
9	32.26	50.12	37.14	66.67	48.89	89.49	**89.93**	86.27	88.89
10	73.84	81.32	66.03	87.77	86.06	97.48	93.94	97.94	97.71
11	84.36	86.91	82.49	89.09	97.51	98.16	97.45	**99.50**	98.69
12	42.89	78.43	73.49	63.67	74.91	93.07	93.18	94.57	98.13
13	98.58	96.35	99.30	100.00	99.46	98.59	**99.12**	94.59	97.28
14	94.02	93.91	93.78	95.33	99.74	99.72	99.39	99.29	**99.91**
15	42.65	77.36	55.39	66.76	84.10	93.31	84.26	92.35	**98.84**
16	92.19	84.38	81.54	91.57	93.98	93.79	89.69	**97.98**	95.54
OA(%)	76.39	83.69	79.37	84.47	91.03	94.78	94.90	96.62	**97.47**
AA(%)	72.18	76.53	70.87	77.83	82.18	94.67	93.85	95.66	**96.57**
$\kappa\times100$	72.85	81.40	76.28	82.24	89.68	94.08	94.17	96.29	**97.11**

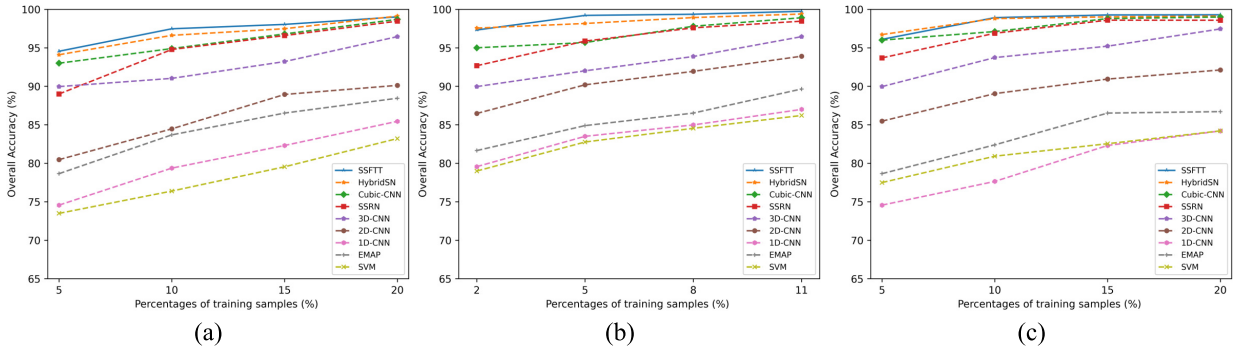

Fig. 10. OA of different models with different training samples' percentages. (a) Indian Pines dataset. (b) Pavia University dataset. (c) Houston 2013 dataset.

Fig. 7 shows the influence of the mutual effect of patch sizes and the number of tokens on the classification accuracy of the three datasets. Patch sizes in the range of $[9, 17]$ and token quantities of $[2, 12]$ were selected. It can be seen from Fig. 7 that the patch size and the number of tokens have an impact on the classification accuracy, and the relationship between the patch size, the number of tokens, and OA cannot be simply expressed by a convex function. The main reason may be that the diversity of semantic information in the square image patches cannot be accurately extracted by the tokens. However, for the Indian pines dataset when the patch size lies in the range of [11, 15] and the number of tokens lies in the range of [8, 12], a locally smooth convex function can also be obtained. This phenomenon shows that setting the suitable patch size and the number of tokens can obtain a relatively smooth OA value.

Table II shows the influence of patch size change on precision when the number of tokens was 4. Since the number of tokens is fixed, the table shows an optimal value for accuracy with patch size, that is, 13×13 for the Indian Pines and Pavia University datasets, and 9×9 for the Houston 2013 dataset.

The influences of the numbers of 3-D convolution kernels and 2-D convolution kernels on the OA are illustrated in the bar charts of Figs. 8 and 9. Fig. 8 shows the experimental data with 64 2-D kernels, and Fig. 9 shows the data with eight 3-D kernels. From the perspective of the Indian Pines dataset, the classification accuracy decreased with more 3-D kernels and increased with more 2-D kernels. Also, the maximum value was reached with eight 3-D kernels and 64 2-D kernels. For the Pavia University dataset, the optimal values were eight 3-D kernels and 32 2-D kernels. In the Houston 2013 dataset, the results were better when the 3-D kernels quantity was 24, and the 2-D kernel quantity was 48. The stability of the overall parameter experimental data indicates that the sensitivity of the classification accuracy to the number of convolution kernels is not significant.

4) Ablation Experiments: To fully demonstrate the effectiveness of the proposed method, ablation experiments were carried out according to the combination of different components on the Pavia University dataset. Five combinations were considered. The influence of different components on the whole model was analyzed in terms of classification accuracy. All experimental results are listed in Table III. In detail, the whole model was divided into four components—3-D convolution layer, 2-D convolution layer, tokenizer, and TE. The model without a 2-D convolution layer yielded the worst classification accuracy. Without a 3-D convolution layer, the model performed better than the first case. In the third case,

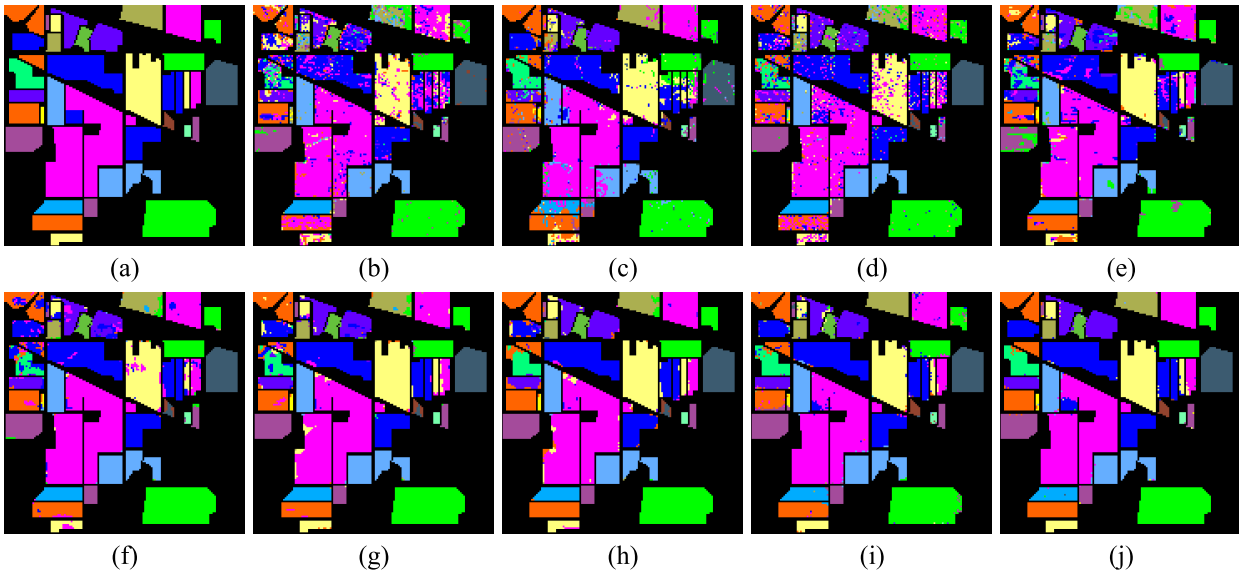

Fig. 11. Classification maps of the Indian Pines dataset. (a) Ground-truth map. (b) SVM (OA = 76.39%). (c) EMAP (OA = 83.69%). (d) 1-D-CNN (OA = 79.37%). (e) 2D-CNN (OA = 84.47%). (f) 3-D-CNN (OA = 91.03%). (g) SSRN (OA = 94.78%). (h) Cubic-CNN (OA = 94.90%). (i) HybridSN (OA = 96.62%). (j) SSFTT (OA = 97.47%).

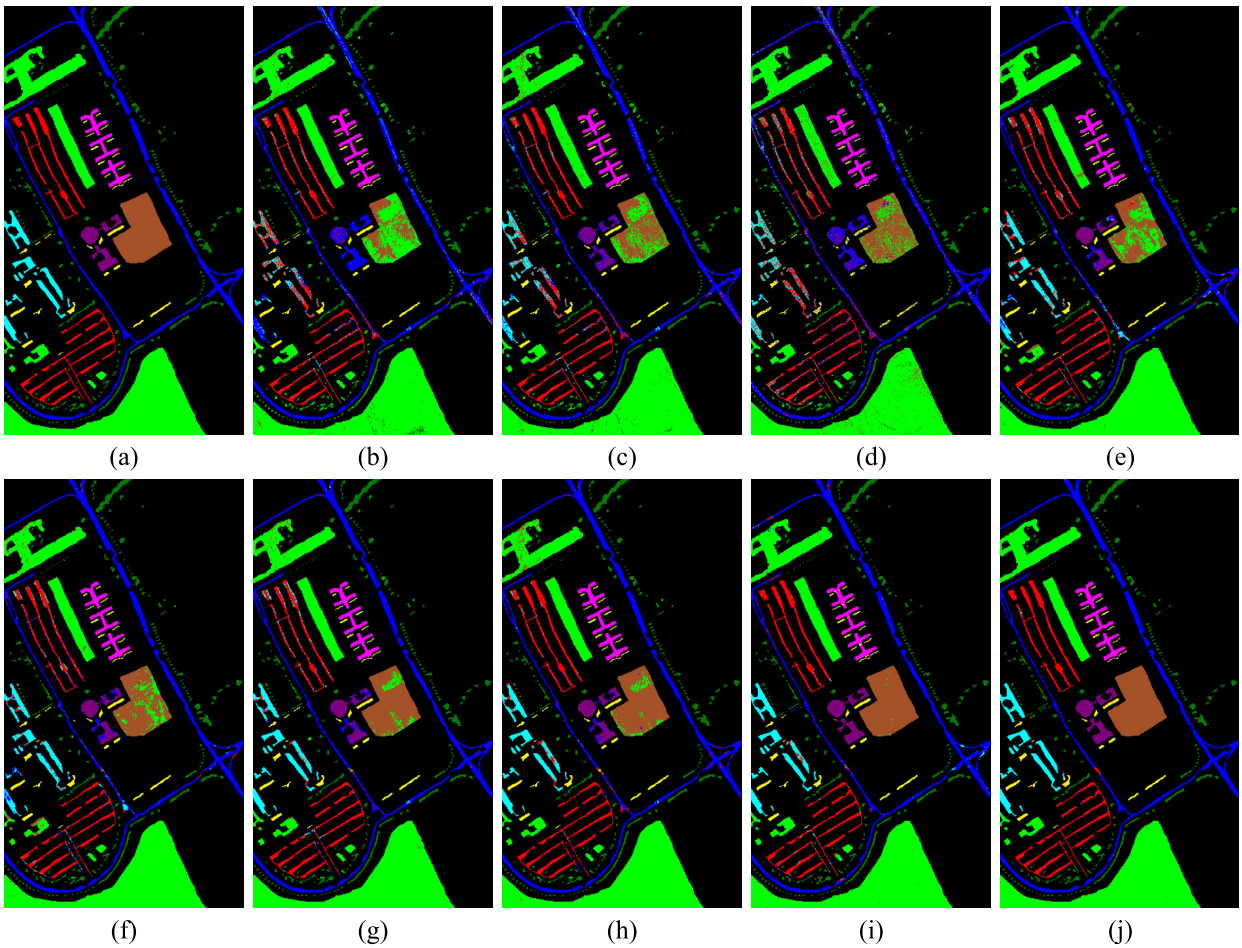

Fig. 12. Classification maps of the Pavia University dataset. (a) Ground-truth map. (b) SVM (OA = 82.76%). (c) EMAP (OA = 84.89%). (d) 1-D-CNN (OA = 83.50%). (e) 2-D-CNN (OA = 90.19%). (f) 3-D-CNN (OA = 92.01%). (g) SSRN (OA = 95.87%). (h) Cubic-CNN (OA = 95.68%). (i) HybridSN (OA = 98.16%). (j) SSFTT (OA = 99.21%).

neither convolution layer was used. However, the patch embedding (PE) method in ViT [55] was introduced to use the transformer structure. The data after PCA dimension reduction were directly used in the PE operation. This case obtained

TABLE V
CLASSIFICATION PERFORMANCE OBTAINED BY DIFFERENT METHODS FOR THE PAVIA UNIVERSITY DATASET (OPTIMAL RESULTS ARE BOLDED)

NO.	SVM	EMAP [30]	1D-CNN [36]	2D-CNN [37]	3D-CNN [40]	SSRN [42]	Cubic-CNN [45]	HybridSN [41]	SSFTT
1	91.22	89.74	84.91	93.28	93.30	95.35	94.48	95.51	**99.33**
2	97.78	93.50	94.14	94.90	93.99	94.69	93.96	99.49	**99.92**
3	34.95	42.06	47.38	75.55	90.19	96.48	93.49	94.18	**98.29**
4	81.55	81.54	82.28	93.87	91.29	96.37	98.01	**99.55**	98.49
5	98.59	97.96	99.76	97.98	95.47	99.69	**99.76**	96.71	99.53
6	41.50	48.18	77.67	70.05	93.85	97.49	93.48	99.43	**100.00**
7	16.71	31.58	19.40	70.92	81.45	95.36	96.48	**100.00**	99.13
8	89.74	91.45	70.01	90.30	92.73	91.49	92.51	95.97	**98.05**
9	**99.89**	98.78	98.55	97.89	95.46	95.90	95.35	95.22	95.44
OA(%)	82.76	84.89	83.50	90.19	92.01	95.87	95.68	98.16	**99.21**
AA(%)	72.44	76.36	74.90	87.31	90.45	95.86	95.28	97.35	**98.69**
$\kappa\times100$	76.28	80.41	77.90	87.52	90.87	95.78	95.55	97.57	**99.15**

an accuracy of 90.98%. In the fourth case, PE was used instead of the tokenizer, preserving two convolution layers. The classification accuracy reached 97.51% in this case, which is considered relatively good, but this value was slightly lower than our proposed method. Therefore, the model using the tokenizer improved the classification effect. The fifth case achieved the classification result of spectral–spatial features obtained only by two convolution layers, with an accuracy of 93.52%. This illustrates that the processing of features by the TE module contributed to performance improvement. In conclusion, the analysis of the combined experimental results further confirmed the validity of our model.

C. Classification Results

To demonstrate the effectiveness of the proposed model, several representative methods were selected for comparative experiments: SVM, EMAP [30], 1-D-CNN [36], 2-D-CNN [37], 3-D-CNN [40], SSRN [42], Cubic-CNN [45], HybridSN [41], and the proposed SSFTT method.

For the SVM, the radial basis kernel function (RBF) was selected for the classification task. In RBF, after cross-validation in the range of $\sigma = [2^{-3}, 2^{-2}, \ldots, 2^4]$ and $\lambda = [10^{-2}, 10^{-1}, \ldots, 10^4]$, an optimal combination of hyperparameters σ and λ was set. For the EMAP, the number of principal components was set to 3. Three attribute profiles were selected, including the area of the region a, the moment of inertia i, and standard deviation std. The scale of the three attribute profiles was set to $a = [100, 500, 1000, 5000]$, $i = [0.2, 0.3, 0.4, 0.5]$, and std $= [20, 30, 40, 50]$. The settings of classifier parameters were identical to those of the SVM above.

For the 1-D-CNN, the network contained five weighted layers, including the input layer, the convolutional layer, the maximum pooling layer, the fully connected layer, and the output layer. The convolution layer contained 20 1-D convolutional filters with an output size of 128. A softmax function was finally added to the top layer of the 1-D-CNN. The 2-D-CNN architecture had three 2-D convolutional blocks and two linear layers. Each convolutional block of 2-D-CNN consisted of a 2-D convolutional layer, a batch normalization (BN) layer, and an ReLU activation function. Each 2-D convolutional layer contained eight, 16, and 32 2-D filters with the size of 3×3, respectively. For the 3-D-CNN, three 3-D convolutional blocks and two linear layers were contained in the network. Similar to the 2-D-CNN, each convolutional block of 3-D-CNN consisted of a 3-D convolutional layer, a BN layer, and an ReLU activation function. Each 3-D convolutional layer contained eight, 16, and 32 3-D filters with the size of $3 \times 3 \times 3$, respectively. For the SSRN, Cubic-CNN, and HybridSN, all network settings were as described in their corresponding references. For the proposed SSFTT method, the number of principal components was set to 30. The 3-D convolution layer was made up of eight kernels of size $3\times3\times3$. The 2-D convolution layer consisted of 64 kernels of size 3×3. The patch width and height $s \times s$ were set to 13×13. Each sample token's number quantity was 4, and the number of TE heads was 4. For a fair comparison, the training sample sets and test sample sets of all methods were randomly selected, as shown in Table I.

1) Quantitative Analysis: Tables IV–VI report the OA, AA, κ, and each class accuracy using all the mentioned methods for the Indian Pines, Pavia University, and Houston 2013 datasets, respectively. The optimal results are denoted in bold. The evaluation data clearly show that the proposed SSFTT method performs the best. The SSFTT obtained the highest OA, AA, κ values, and classification accuracy values. Specifically, Table IV shows that there were several unbalanced classification results in detail. For example, categories "alfalfa," "grass-pasture-mowed," and "oats" have poor results on separate classes in SVM, EMAP, and 1-D-CNN because the sample sizes of these categories were relatively small. In addition, the random sampling was based on percentage sampling, resulting in a relatively small training number of these categories and sample imbalance. However, the accuracy of each class obtained by SSFTT was relatively uniform, which validates the validity of extending the classification of small samples to regional semantic concepts. The same condition was evident in Tables V and VI.

In the Pavia University and Houston 2013 datasets, "gravel," "bare soil," and "parking lot 2" perform poorly in the first two methods. In another case, the classification accuracies of many single categories of HybridSN are close to or even higher than that of the SSFTT in Table VI. The main reason is due to the Houston 2013 dataset, in which the sample points are mostly discrete and local, unlike the first two datasets, where most areas belong to the same category. Therefore, our method does not have enough advantage in semantic modeling on this dataset, but it did achieve higher results.

TABLE VI

CLASSIFICATION PERFORMANCE OBTAINED BY DIFFERENT METHODS FOR THE HOUSTON 2013 DATASET (OPTIMAL RESULTS ARE BOLDED)

NO.	SVM	EMAP [30]	1D-CNN [36]	2D-CNN [37]	3D-CNN [40]	SSRN [42]	Cubic-CNN [45]	HybridSN [41]	SSFTT
1	95.65	87.85	86.20	94.02	93.75	94.98	94.46	**98.85**	98.84
2	97.52	92.53	95.13	96.30	94.91	95.60	96.65	**99.73**	99.38
3	99.84	99.84	**100.00**	89.73	95.54	97.14	96.84	99.84	99.52
4	93.66	92.63	95.43	98.31	94.91	99.45	94.56	96.07	**98.39**
5	98.30	97.26	98.98	97.88	**100.00**	99.80	99.83	**100.00**	**100.00**
6	84.25	82.16	95.15	73.46	89.86	94.98	94.59	**100.00**	**100.00**
7	82.65	80.47	76.60	89.63	91.84	88.49	93.48	**97.63**	96.67
8	56.61	70.51	58.12	80.96	80.36	95.86	96.49	**97.95**	97.68
9	73.91	72.69	63.16	70.98	93.58	92.78	91.21	98.67	**99.29**
10	83.51	86.78	55.57	84.65	94.28	96.49	95.89	**99.00**	98.73
11	68.97	64.59	70.16	90.96	92.64	97.85	97.98	99.28	**99.91**
12	60.72	59.18	54.74	91.46	94.37	99.32	98.12	99.46	**99.55**
13	25.69	45.29	41.93	85.65	90.97	92.69	96.36	98.10	**98.10**
14	93.25	96.48	97.05	73.71	94.58	98.46	97.46	**100.00**	**100.00**
15	**99.66**	98.45	99.04	99.52	99.65	99.02	99.12	99.49	99.16
OA(%)	80.92	82.39	77.64	89.05	93.73	96.89	97.11	98.80	**98.92**
AA(%)	79.61	78.49	79.15	87.82	93.56	96.19	96.23	98.93	**99.01**
$\kappa\times100$	79.33	80.64	75.81	88.15	93.63	96.54	96.68	98.71	**98.83**

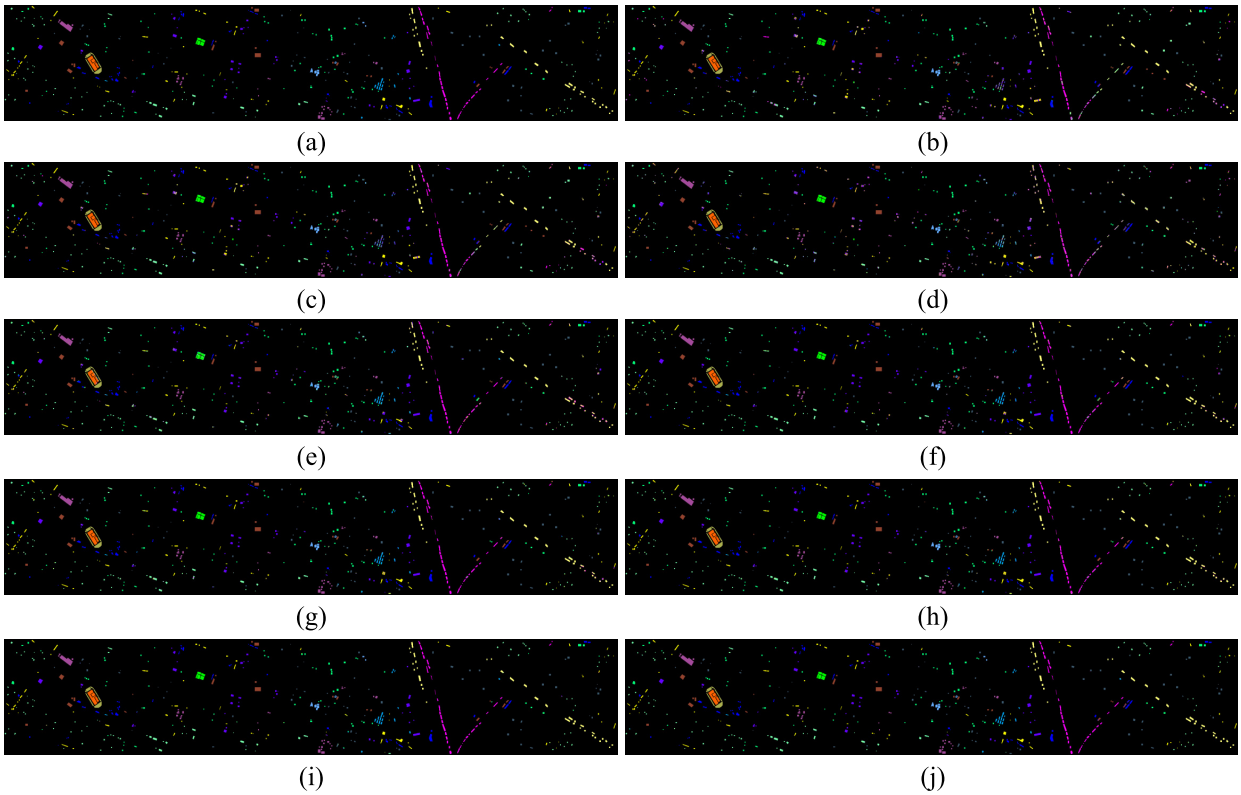

Fig. 13. Classification maps of the Houston 2013 dataset. (a) Ground-truth map. (b) SVM (OA = 80.92%). (c) EMAP (OA = 82.39%). (d) 1-D-CNN (OA = 77.64%). (e) 2-D-CNN (OA = 89.05%). (f) 3-D-CNN (OA = 93.73%). (g) SSRN (OA = 96.89%). (h) Cubic-CNN (OA = 97.11%). (i) HybridSN (OA = 98.80%). (j) SSFTT (OA = 98.92%).

Fig. 10 shows the classification accuracies of several methods with different proportions of training samples. Considering the stability and robustness of the proposed method under different percentages of training samples, 5%, 10%, 15%, and 20% labeled samples were randomly selected as training data for the Indian Pines and Houston 2013 datasets and 2%, 5%, 8%, and 11% for the Pavia University dataset in the experiment. In the case of a small number of samples, our method still resulted in a good performance. In addition, when the sample proportion was 10%, all the other methods were less accurate for each dataset. However, as the number of samples increased, the HybridSN method performed almost as well or even better than our method. The accuracy does not indicate a significant difference because the accuracy is almost 100%.

At the same time, to further verify the effectiveness of the proposed method, two latest transformer-based methods were selected for comparison: SST [56] and SF [59]. Since

TABLE VII

EXPERIMENTAL RESULTS BETWEEN THE TRANSFORMER-BASED METHODS AND THE PROPOSED METHOD

	Salinas		Pavia University		Indian Pines		Pavia University	
Methods	SSFTT	SST [56]	SSFTT	SST [56]	SSFTT	SF [59]	SSFTT	SF [59]
OA(%)	95.31	94.42	92.83	92.74	96.83	81.76	99.89	91.07
AA(%)	96.98	93.11	88.19	83.60	93.20	87.81	99.67	90.02
$\kappa\times100$	94.78	93.73	90.45	90.80	96.39	79.19	99.85	88.05
Train. time	14.02s	22.42m	14.6s	16.69m	-	-	-	-

* The results of SST and SF networks are directly chosen from refrences [56] and [59], respectively.

TABLE VIII

TRAINING TIME IN MINUTES (min) AND TEST TIME IN SECONDS (s) BETWEEN THE CONTRAST METHODS AND THE PROPOSED METHOD ON THREE DATASETS (OPTIMAL RESULTS ARE BOLDED, AND SUBOPTIMAL RESULTS ARE UNDERLINED)

Methods	Indian Pines		Pavia University		Houston 2013	
	Train.(m)	Test (s)	Train.(m)	Test (s)	Train.(m)	Test (s)
2D-CNN [37]	**1.45**	**3.32**	**1.91**	**4.82**	**1.19**	**3.34**
3D-CNN [40]	7.26	6.90	11.48	14.62	8.29	9.10
SSRN [42]	11.74	8.83	14.46	24.43	8.93	11.75
Cubic-CNN [45]	10.54	9.60	13.58	29.91	9.73	15.18
HybridSN [41]	6.61	7.93	9.32	28.40	6.73	8.49
SSFTT	<u>2.29</u>	<u>5.29</u>	<u>2.17</u>	<u>9.85</u>	<u>1.62</u>	<u>3.43</u>

there is no public code for these two methods, we set the same selection of training samples and test samples in the comparison experiments according to the references and compared the results of the proposed SSFTT with the results given in the references of these two methods. Specifically, in the comparison experiments of the SST method, the Salinas[2] dataset and the Pavia University dataset were used, and totally 200 samples in each dataset were selected as training samples. Table VII shows the comparison results. It can be seen from the comparison result with the SST method that the proposed SSFTT is slightly better than the SST method in classification performance, but it consumes much less time. For example, on the Salinas dataset, the OA index of SSFTT is 0.89% higher than that of SST, but the training time of the SSFTT network is only 14.02 s, which is much lower than the training time of the SST network of 22.42 min. Compared with the SF method, the proposed SSFTT method surpasses the SF method with an overwhelming advantage in OA, AA, and κ indicators.

2) Visual Evaluation: The classification maps of several comparison methods are shown in Figs. 11–13 for the Indian Pines, Pavia University, and Houston 2013 datasets, respectively. By visual comparison, the classification map obtained by SSFTT is the cleanest and the closest to the ground-truth map. The first few methods of SVM, EMAP, and 1-D-CNN clearly contain significant noise for the three datasets. Indirectly, these models cannot accurately identify the types of objects, and their performances are poor. For the Indian Pines dataset, the comparison shows that the small blue square in the middle of each image is a relatively indistinguishable area, and almost all the comparison methods are wrongly classified as pink. Our method identifies the blue areas to a greater extent, which also demonstrates the excellent performance of our method. For the Pavia University dataset, the category "bare soil" in brown is more prominent. In this category, several other comparison methods exhibit considerable green clutter. However, this category looks clean on the SSFTT classification map. Similarly, for the Houston 2013 dataset, by comparing some of the boundary regions shown in Fig. 13, it is concluded that our method maintains the best boundary regions, which further verifies the classification performance of our method.

3) Time Cost Comparison: A comparison of training and testing time for 2-D-CNN, 3-D-CNN, SSRN, Cubic-CNN, HybridSN, and our method is shown in Table VIII. It is clear that the SSFTT method is relatively faster than the other methods except for 2-D-CNN. Therefore, our method can effectively reduce the computation time and improve classification efficiency. The training and testing processes of the SSRN and Cubic-CNN take a relatively long time because their network layers are deeper than other structures, and each iteration consumes a large calculation cycle. The SSFTT takes slightly longer than 2-D-CNN because of the use of a 3-D convolution layer to extract spectral features in this framework. In addition, the time for the residual calculation in the TE structure is significant.

[2]http://www.ehu.eus/ccwintco/index.php?title=Hyperspectral_Remote_Sensing_Scenes#Salinas_scene

IV. CONCLUSION

This article proposes an SSFTT method for improving the performance of HSI classification. This method integrates a backbone CNN and transformer structure organically. The convolution layers are used to fully capture low-level convolution spectral–spatial features. Then, the features are transformed into semantic tokens. In addition, the TE structure is used to model the high-level semantic features of tokens. Such an operation makes the analysis of land-cover characteristics more sufficient. This method is shown to improve classification performance effectively and efficiently in the experiments. Furthermore, the validity of extending the HSI classification problem to the local high-level semantic classification problem is demonstrated.

In the future, based on the lightweight SSFTT, we will study an end-to-end transformer network with two branches in spatial and spectral domains for extracting high-level spatial–spectral features, thereby further improving the classification accuracy. In addition, the proposed SSFTT has good scalability in the high-level semantic feature extraction of multimodal data, so as to provide a new idea for network design of the joint fusion and classification of hyperspectral and LiDAR data.

REFERENCES

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

[2] 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, Jun. 2015.

[3] S. S. M. Noor, K. Michael, S. Marshall, J. Ren, J. Tschannerl, and F. Kao, "The properties of the cornea based on hyperspectral imaging: Optical biomedical engineering perspective," in *Proc. Int. Conf. Syst., Signals Image Process. (IWSSIP)*, May 2016, pp. 1–4.

[4] J. Wang, L. Zhang, Q. Tong, and X. Sun, "The spectral crust project—Research on new mineral exploration technology," in *Proc. 4th Workshop Hyperspectral Image Signal Process., Evol. Remote Sens. (WHISPERS)*, Jun. 2012, pp. 1–4.

[5] A. Fong, G. Shu, and B. McDonogh, "Farm to table: Applications for new hyperspectral imaging technologies in precision agriculture, food quality and safety," in *Proc. Conf. Lasers Electro-Optics (CLEO)*, 2020, pp. 1–2.

[6] J.-P. Ardouin, J. Lévesque, and T. A. Rea, "A demonstration of hyperspectral image exploitation for military applications," in *Proc. 10th Int. Conf. Inf. Fusion*, Jul. 2007, pp. 1–8.

[7] L. Sun, C. He, Y. Zheng, and S. Tang, "SLRL4D: Joint restoration of subspace low-rank learning and non-local 4-D transform filtering for hyperspectral image," *Remote Sens.*, vol. 12, no. 18, p. 2979, Sep. 2020.

[8] C. He, L. Sun, W. Huang, J. Zhang, Y. Zheng, and B. Jeon, "TSLRLN: Tensor subspace low-rank learning with non-local prior for hyperspectral image mixed denoising," *Signal Process.*, vol. 184, Jul. 2021, Art. no. 108060.

[9] L. Sun, F. Wu, T. Zhan, W. Liu, J. Wang, and B. Jeon, "Weighted nonlocal low-rank tensor decomposition method for sparse unmixing of hyperspectral images," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 1174–1188, 2020.

[10] S. Yang and Z. Shi, "Hyperspectral image target detection improvement based on total variation," *IEEE Trans. Image Process.*, vol. 25, no. 5, pp. 2249–2258, May 2016.

[11] L. Sun, Z. Wu, J. Liu, L. Xiao, and Z. Wei, "Supervised spectral–spatial hyperspectral image classification with weighted Markov random fields," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 3, pp. 1490–1503, Mar. 2015.

[12] L. Sun *et al.*, "Low rank component induced spatial–spectral kernel method for hyperspectral image classification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 10, pp. 3829–3842, Oct. 2020.

[13] L. Sun, C. Ma, H. J. Shim, Z. Wu, and B. Jeon, "Adjacent superpixel-based multiscale spatial–spectral kernel for hyperspectral classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 12, no. 6, pp. 1905–1919, Jun. 2019.

[14] C. Cariou and K. Chehdi, "A new k-nearest neighbor density-based clustering method and its application to hyperspectral images," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2016, pp. 6161–6164.

[15] Y. E. SahIn, S. Arisoy, and K. Kayabol, "Anomaly detection with Bayesian Gauss background model in hyperspectral images," in *Proc. 26th Signal Process. Commun. Appl. Conf. (SIU)*, May 2018, pp. 1–4.

[16] J. Haut, M. Paoletti, A. Paz-Gallardo, J. Plaza, and A. Plaza, "Cloud implementation of logistic regression for hyperspectral image classification," in *Proc. 17th Int. Conf. Comput. Math. Methods Sci. Eng. (CMMSE)*, vol. 3. Cádiz, Spain: Costa Ballena, Jul. 2017, pp. 1063–2321.

[17] J. Li, J. Bioucas-Dias, and A. Plaza, "Spectral–spatial hyperspectral image segmentation using subspace multinomial logistic regression and Markov random fields," *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 3, pp. 809–823, Aug. 2012.

[18] 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.

[19] Q. Ye, P. Huang, Z. Zhang, Y. Zheng, L. Fu, and W. Yang, "Multiview learning with robust double-sided twin SVM," *IEEE Trans. Cybern.*, early access, Sep. 21, 2021, doi: 10.1109/TCYB.2021.3088519.

[20] Q. Ye *et al.*, "L1-norm distance minimization-based fast robust twin support vector k-plane clustering," *IEEE Trans. neural Netw. Learn. Syst.*, vol. 29, no. 9, pp. 4494–4503, Sep. 2017.

[21] Y.-N. Chen, T. Thaipisutikul, C.-C. Han, T.-J. Liu, and K.-C. Fan, "Feature line embedding based on support vector machine for hyperspectral image classification," *Remote Sens.*, vol. 13, no. 1, p. 130, Jan. 2021.

[22] G. Licciardi, P. R. Marpu, J. Chanussot, and J. A. Benediktsson, "Linear versus nonlinear PCA for the classification of hyperspectral data based on the extended morphological profiles," *IEEE Geosci. Remote Sens. Lett.*, vol. 9, no. 3, pp. 447–451, May 2012.

[23] S. Prasad and L. M. Bruce, "Limitations of principal components analysis for hyperspectral target recognition," *IEEE Geosci. Remote Sens. Lett.*, vol. 5, no. 4, pp. 625–629, Oct. 2008.

[24] A. Villa, J. A. Benediktsson, J. Chanussot, and C. Jutten, "Hyperspectral image classification with independent component discriminant analysis," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 12, pp. 4865–4876, Dec. 2011.

[25] Q. Ye, J. Yang, F. Liu, C. Zhao, N. Ye, and T. Yin, "L1-norm distance linear discriminant analysis based on an effective iterative algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 1, pp. 114–129, Jan. 2018.

[26] L. Fu *et al.*, "Learning robust discriminant subspace based on joint $L_{2,p}$- and $L_{2,s}$-norm distance metrics," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 1, pp. 130–144, Jan. 2022.

[27] T. V. Bandos, L. Bruzzone, and G. Camps-Valls, "Classification of hyperspectral images with regularized linear discriminant analysis," *IEEE Trans. Geosci. Remote Sens.*, vol. 47, no. 3, pp. 862–873, Mar. 2009.

[28] 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. 2007.

[29] J. A. Benediktsson, J. A. Palmason, and J. R. Sveinsson, "Classification of hyperspectral data from urban areas based on extended morphological profiles," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 3, pp. 480–491, Mar. 2005.

[30] M. M. Dalla, A. Villa, J. A. Benediktsson, J. Chanussot, and L. Bruzzone, "Classification of hyperspectral images by using extended morphological attribute profiles and independent component analysis," *IEEE Geosci. Remote Sens. Lett.* vol. 8, no. 3, pp. 542–546, Dec. 2011.

[31] Y. Duan, H. Huang, and T. Wang, "Semisupervised feature extraction of hyperspectral image using nonlinear geodesic sparse hypergraphs," *IEEE Trans. Geosci. Remote Sens.*, early access, Sep. 15, 2021, doi: 10.1109/TGRS.2021.3110855.

[32] F. Luo, Z. Zou, J. Liu, and Z. Lin, "Dimensionality reduction and classification of hyperspectral image via multi-structure unified discriminative embedding," *IEEE Trans. Geosci. Remote Sens.*, early access, Nov. 16, 2021, doi: 10.1109/TGRS.2021.3128764.

[33] M. E. Paoletti, J. M. Haut, J. Plaza, and A. Plaza, "Deep learning classifiers for hyperspectral imaging: A review," *ISPRS J. Photogramm. Remote Sens.*, vol. 158, pp. 279–317, Dec. 2019.

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

[35] Y. S. 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. 2014.

[36] W. Hu, Y. Huang, L. Wei, F. Zhang, and H. Li, "Deep convolutional neural networks for hyperspectral image classification," *J. Sensors*, vol. 2015, pp. 1–12, Jan. 2015.

[37] W. Shao and S. Du, "Spectral–spatial feature extraction for hyperspectral image classification: A dimension reduction and deep learning approach," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 8, pp. 4544–4554, Oct. 2016.

[38] J. Yue, W. Zhao, S. Mao, and H. Liu, "Spectral–spatial classification of hyperspectral images using deep convolutional neural networks," *Remote Sens. Lett.*, vol. 6, no. 6, pp. 468–477, May 2015.

[39] 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.*, vol. 55, no. 8, pp. 4729–4742, Aug. 2017.

[40] 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.

[41] S. K. Roy, G. Krishna, S. R. Dubey, and B. B. Chaudhuri, "HybridSN: Exploring 3-D–2-D CNN feature hierarchy for hyperspectral image classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 17, no. 2, pp. 277–281, Feb. 2020.

[42] 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, Aug. 2018.

[43] M. E. Paoletti, J. M. Haut, R. Fernandez-Beltran, J. Plaza, A. J. Plaza, and F. Pla, "Deep pyramidal residual networks for spectral–spatial hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 2, pp. 740–754, Aug. 2019.

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

[45] J. Wang, X. Song, L. Sun, W. Huang, and J. Wang, "A novel cubic convolutional neural network for hyperspectral image classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 4133–4148, 2020.

[46] Q. Liu, Z. Wu, Q. Du, Y. Xu, and Z. Wei, "Multiscale alternately updated clique network for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–15, Jun. 2021, doi: 10.1109/TGRS.2021.3090413.
[47] J. Li, X. Zhao, Y. Li, Q. Du, B. Xi, and J. Hu, "Classification of hyperspectral imagery using a new fully convolutional neural network," *IEEE Geosci. Remote Sens. Lett.*, vol. 15, no. 2, pp. 292–296, Feb. 2018.
[48] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez, "Fully convolutional neural networks for remote sensing image classification," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2016, pp. 5071–5074.
[49] R. Hang, Q. Liu, D. Hong, and P. Ghamisi, "Cascaded recurrent neural networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 8, pp. 5384–5394, Aug. 2019.
[50] H. Wu and S. Prasad, "Convolutional recurrent neural networks for hyperspectral data classification," *Remote Sens.*, vol. 9, no. 3, p. 298, Mar. 2017.
[51] 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, Sep. 2018.
[52] Y. Zhan, D. Hu, Y. Wang, and X. Yu, "Semisupervised hyperspectral image classification based on generative adversarial networks," *IEEE Geosci. Remote Sens. Lett.*, vol. 15, no. 2, pp. 212–216, Feb. 2018.
[53] M. E. Paoletti *et al.*, "Capsule networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 4, pp. 2145–2160, Apr. 2019.
[54] D. Hong, L. Gao, J. Yao, B. Zhang, A. Plaza, and J. Chanussot, "Graph convolutional networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 7, pp. 5966–5978, Jul. 2021.
[55] A. Dosovitskiy *et al.*, "An image is worth 16×16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.
[56] X. He, Y. Chen, and Z. Lin, "Spatial–spectral transformer for hyperspectral image classification," *Remote Sens.*, vol. 13, no. 3, p. 498, Jan. 2021.
[57] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
[58] Y. Qing, W. Liu, L. Feng, and W. Gao, "Improved transformer net for hyperspectral image classification," *Remote Sens.*, vol. 13, no. 11, p. 2216, Jun. 2021.
[59] D. Hong *et al.*, "SpectralFormer: Rethinking hyperspectral image classification with transformers," 2021, *arXiv:2107.02988*.
[60] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

Le Sun (Member, IEEE) was born in Jiangsu, China, in 1987. He received the B.S. degree from the School of Science, Nanjing University of Science and Technology (NJUST), Nanjing, Jiangsu, China, in 2009, and the Ph.D. degree from the School of Computer Science and Engineering, NJUST, in 2014.

From 2015 to 2018, he conducted research in the field of multi-images fusion based on sparse dictionary learning and compressive sensing as a Post-Doctoral Researcher with the School of Electronic and Electrical Engineering, Sungkyunkwan University, Seoul, South Korea. Since 2020, he has been an Associate Professor with the School of Computer and Science, Nanjing University of Information Science and Technology, Nanjing. His research interests include hyperspectral image processing (including unmixing, classification, and restoration), sparse representation, compressive sensing, and deep learning.

Guangrui Zhao received the B.S. degree in mathematics and applied mathematics from the School of Mathematics and Statistics, Qingdao University, Qingdao, China, in 2019. He is currently pursuing the M.S. degree in software engineering with the Nanjing University of Information Science and Technology, Nanjing, China, in 2020.

His research interest includes hyperspectral image processing.

Yuhui Zheng (Member, IEEE) was born in Shanxi, China, in 1982. He received the B.S. degree in chemistry and the Ph.D. degree in computer science from the Nanjing University of Science and Technology (NJUST), Nanjing, Jiangsu, in 2004 and 2009, respectively.

From 2014 to 2015, he was a Visiting Scholar with the Digital Media Laboratory, School of Electronic and Electrical Engineering, Sungkyunkwan University, Seoul, South Korea. He is currently a Full Professor with the School of Computer and Science, Nanjing University of Information Science and Technology, Nanjing. His research interests cover image processing, pattern recognition, and remote sensing information systems.

Zebin Wu (Senior Member, IEEE) was born in Zhejiang, China, in 1981. He received the B.S. and Ph.D. degrees from the School of Computer Science and Engineering, Nanjing University of Science and Technology (NJUST), Nanjing, Jiangsu, China, in 2003 and 2008, respectively.

He is currently a Full Professor with the School of Computer Science and Engineering, NJUST. His research interests include virtual reality and system simulation, remote sensing information processing, and distributed computing.