Class-Level Joint Sparse Representation for Multifeature-Based Hyperspectral Image Classification

Erlei Zhang, Licheng Jiao, *Senior Member, IEEE*, Xiangrong Zhang, *Senior Member, IEEE*, Hongying Liu, *Member, IEEE*, and Shuang Wang, *Member, IEEE*

***Abstract*—Recent studies show that different features can represent different characteristics of hyperspectral images, and a combination of them would have positive influence on classification. In this paper, we formulate the multifeature hyperspectral image classification as a joint sparse representation model which simultaneously represents the pixels of multiple features (spectral, shape, and texture) with a class-level sparse constraint. The proposed model enforces pixels in a small region of each type features to share the same sparsity pattern; at the same time, the pixels described by different features have freedom to adaptively choose their own appropriate atoms but still belong to the same class. Thus, the proposed model not only preserves the spatial information by joint sparse constraint but also utilizes additional complementary information from different features by class-level sparse constraint. Furthermore, we also kernelize the model to handle nonlinearity in the data. And a new version of simultaneous orthogonal matching pursuit is proposed to solve the aforementioned problems. Experiments on several real hyperspectral images indicate that the proposed algorithms provide a competitive performance when compared with several state-of-the-art algorithms.**

***Index Terms*—Class-level sparsity pattern, feature extraction, hyperspectral imagery (HSI) classification, multifeature, orthogonal matching pursuit.**

I. Introduction

HYPERSPECTRAL imagery (HSI) is three-dimensional (3-D) optical pattern obtained by imaging spectrometer. Each pixel includes two kinds of information: 1) two-dimensional space coordinates information of pixels and 2) one-dimensional spectral information with a large spectral wavelength range. Every pixel of a HSI can be represented by a vector whose entries offer fine spectral differences among different materials [1]. Therefore, HSI is widely used for military affairs, precision agriculture [2], environmental protection [3], and so on.

Classification is one of the most important tasks of HSI understanding. The key point of classification is to classify every pixel into certain classes by using the meaningful information from the related long vector. Over the past decades, many spectral pixelwise classifiers [4]–[9] have been developed. The basic idea of the pixelwise classification approach is as follows: each pixel is considered as an independent pattern, and its spectrum is considered as the initial feature. These methods commonly make full use of the spectral information for classification.

The spatial-spectral classifiers [10] belong to another kind. Most of these methods are proposed based on the assumption that pixels in a local region usually have similar spectral characteristics. There are several ways to put the spatial information into classification, including using the postprocessing procedure [11], [12], constructing composite kernel [13], [14], adding Bayesian-based regular terms [15], [16], and joint sparsity model [17], [18]. The classification performance of the spatial-spectral approaches can be improved by incorporating the spatial information of HSI. These approaches can alleviate the noise, and the obtained classification maps are often smooth.

Recently, sparse representation has drawn greater attention in the classification of HSI [17]–[21]. In classification, a test pixel can be approximated using a linear combination of a few dictionary atoms. The class label of the test pixel can be determined by the minimal reconstruction error. Chen *et al.* [17] proposed a joint sparse representation classifier (SRC) based on the joint sparsity model [22] with neighboring pixel information for HSI classification. Li *et al.* [23] utilized collaborative representation (CR) mechanism [24], [25] to support HSI classification. Even though the above methods have achieved very promising performance, one single type of feature can only depict the HSI from one perspective. It is proved that different feature descriptors provide different discriminative powers [26]–[28]. Inspired by multitask learning theory [29]–[31], some methods have been extended to multiple features learning to solve this problem [32]–[39]. In terms of different norms used in optimizers, the available methods can be roughly grouped into three categories: sparse representation with the $\ell_{1,2}$-norm minimization [32]–[35], sparse representation with the ℓ_2-norm minimization [36], [37], sparse representation with the $\ell_{\text{row},0}$-norm [38]. Specifically, Yuan *et al.* [32] proposed a

Manuscript received February 17, 2015; revised January 16, 2016; accepted January 22, 2016. This work was supported in part by the National Basic Research Program of China (973 Program) under Grant 2013CB329402, in part by the National Natural Science Foundation of China under Grant 61272282, Grant 61203303, Grant 61272279, Grant 61373111, Grant 31300473, and Grant 61501353, in part by the Program for New Century Excellent Talents in University under Grant NCET-13-0948, in part by the Program for New Scientific and Technological Star of Shaanxi Province under Grant 2014KJXX-45, and in part by the China Postdoctoral Science Foundation under Grant 65ZY1425. *(Corresponding author: Xiangrong Zhang).*

The authors are with the Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education, Xidian University, Xi'an 710071, China (e-mail: xrzhang@mail.xidian.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSTARS.2016.2522182

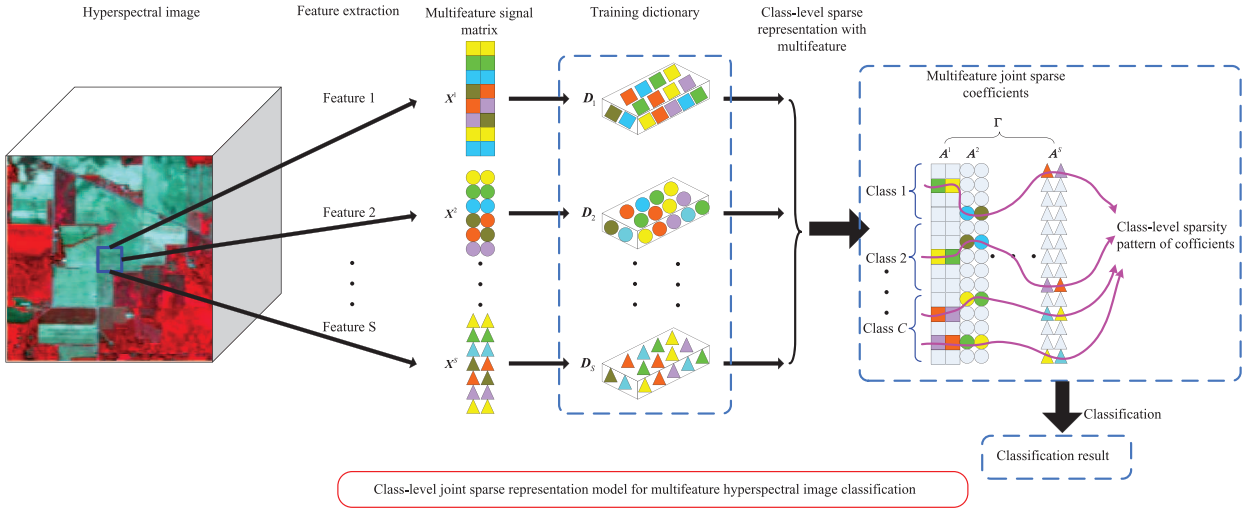


Fig. 1. Illustration of CL-JSRC [see (10)]. In the multifeature joint sparse coefficients, each column represents a sparse coefficient vector corresponding to one pixel; each patch (square, circle, and triangular) in the column is a coefficient value; white blocks denote zero values, whereas color blocks stand for nonzero values. Sparse coefficients of each type features share the same sparsity pattern at atom level, while sparse coefficients of different features have the sparsity pattern at class level, i.e., nonzero coefficients are only associated with training samples belonging to the correct class in the dictionary.

joint SRC with multitask learning (MTJSRC) for face recognition and visual classification. Zheng *et al.* [33] applied MTJSRC with spatial filtering postprocessing into large-scale satellite image annotation. Li *et al.* [35] extended MTJSRC with superpixel segmentation strategy to efficiently utilize spatial information. Li *et al.* [36] proposed a joint CR model with multitask learning (JCRC-MTL) for HSI classification. Furthermore, Li *et al.* [37] generalized JCRC-MTL framework to kernel space by a column-generation-based technique. Zhang *et al.* [38] simultaneously represented pixels of multifeature with an $\ell_{\text{row},0}$-norm regularization. Fang *et al.* [39] proposed a multiscale adaptive sparse representation model to exploit spatial information at multiple scales. All the above representation-based methods skillfully considered the sparsity pattern among the coefficients of multiple input signals, multiple features or multiple scales, and achieved promising classification performances. However, challenges still remain in integrating various features (e.g., spectral feature, shape feature, and texture feature) under the joint sparse representation framework. For instance, the sparse representation with $\ell_{1,2}$-norm minimization methods always need multiple iterations to solve the optimization problem, which is time consuming. Although the sparse representation with the ℓ_2-norm minimization methods leads to less computational complexity than the methods with $\ell_{1,2}$-norm, the model solving process in [36] includes multiple matrix inverse calculations, which is time consuming when dealing with a large dictionary. Thus, developing more efficient and effective methods is essential for future study on sparse representation. The sparse representation with $\ell_{\text{row},0}$-norm minimization method exploits the simultaneous orthogonal matching pursuit (SOMP) technique, which costs less computational time than those $\ell_{1,2}$-norm-based sparse representation algorithms. However, simply enforcing the representation coefficients sharing the same sparsity pattern among all the features is too strict and is not held in practice [25], [39], because there are differences among different features. Fang *et al.* [39] addressed this problem by using the adaptive sparse representation model, while they focused on the selection of the optimal region size and required the dimensions of the inputs be equal, which is inappropriate for multiple features' fusion. Furthermore, if we deal with the classification in kernel space, we can solve the nonlinear separability cases, leading to an enhanced HSI classification performance. He *et al.* [34] have extended kernel MTJSRC to HSI classification, in which they put more emphasis on effectively extracting the spectral-spatial features and only can handle single test input case.

In this paper, we present an alternative joint SRC (JSRC) motivated by the multitask joint sparse representation models [32], [36], [39] for HSI classification. Fig. 1 presents an overview of our framework. We construct a class-level JSRC (CL-JSRC) simultaneously processing multiple features' information and contextual neighborhood knowledge with a class-level sparse constraint. Compared to those previously mentioned methods, our method has some differences and novelties as follows.

1) We present an alternative classifier to fuse multiple features for HSI classification. Under our joint sparse framework, different spectral-spatial features with unequal dimensions can be easily handled and interact through their representation coefficients. Moreover, our algorithm can efficiently handle multiple test inputs, which can be used to encode contextual neighborhood knowledge in the learned model.
2) Instead of the strict requirement of the coefficients being similar or sharing the same sparsity pattern among all the features at atom level, we assume that the coefficients share a class-level sparsity pattern. More specifically, the class-level sparse strategy first enforces pixels in a small region of each type features to share the same sparsity pattern, i.e., the positions of nonzero representation coefficients lie in the same atoms. Then, pixels from different features have freedom to adaptively choose their own appropriate atoms but still belong to the same class. This strategy combines the information of different

features for discrimination and achieves an effective representation for each feature.

3) Nonlinear separability problem is a popular issue to deal with for HSI classification. We kernelize the algorithm to handle nonlinearity in the data, which can improve the classification accuracy.

We use a modified SOMP (MSOMP) to approximately solve this problem with less computational complexity. The modification mainly lies in the greedy selection rule which includes following steps: 1) find the best representation atom for each class and each type of feature; 2) combine the best atoms across the features into a support; and 3) choose the best class subset and update the class-level support set.

This paper is organized as follows. The related works are introduced in Section II. The details of the proposed fast class-level sparsity representation-based method and kernel-view extensions are described in Section III. In Section IV, the efficiency and effectiveness of the proposed methods are demonstrated by experimental results on several real hyperspectral images. Finally, Section V makes a summarization and some closing remarks.

II. Related Work

In this part, we first examine the traditional SRC for single-pixel classification and then review JSRC for incorporating the contextual information.

A. Single-Pixel SRC

Sparse representation-based methods have aroused much concern. Wright *et al.* propose SRC [40] for face recognition. They assume that samples of a certain class should approximately lie in a low-dimensional subspace spanned by the training samples of the same class. In other words, SRC uses a few atoms from a given dictionary to approximately represent a test signal. The objective of SRC is finding out the sparse representation coefficient $\boldsymbol{\alpha} \in \Re^N$ for a testing sample $\boldsymbol{x} \in \Re^b$ in reconstruction. For dictionary $\boldsymbol{D} = [\boldsymbol{D}_1, \ldots, \boldsymbol{D}_c, \ldots, \boldsymbol{D}_C] \in \Re^{b \times N}$ (where $\boldsymbol{D}_c \in \Re^{b \times N_c}$ is the cth class subdictionary whose columns (atoms) are extracted from the cth training samples; b is the dimension of feature vector; C is the number of classes; N_c is the number of atoms in subdictionary $\boldsymbol{D}_c$; and $N = \sum_{c=1}^{C} N_c$ is the total number of atoms in $\boldsymbol{D}$), the representation coefficients $\boldsymbol{\alpha}$ can be obtained through dealing with the following problem:

$$\hat{\boldsymbol{\alpha}} = \arg\min \|\boldsymbol{D}\boldsymbol{\alpha} - \boldsymbol{x}\|_2 \quad \text{s.t.} \quad \|\boldsymbol{\alpha}\|_0 \leq \mathrm{K} \tag{1}$$

where K is a predefined upper bound on the sparsity degree, representing the maximum number of the nonzero coefficients in $\hat{\boldsymbol{\alpha}}$. The essence of above problem is minimizing the reconstruction error within a certain sparsity degree. The class label of the test pixel $\boldsymbol{x}$ is decided by the minimal representation error (between $\boldsymbol{x}$ and its approximation from the subdictionary $\boldsymbol{D}_i$ of each class) when the coefficient vector $\hat{\boldsymbol{\alpha}}$ is obtained, i.e.,

$$\text{label}(\boldsymbol{x}) = \arg \min_{i=1,\ldots,C} r_i(\boldsymbol{x}) = \arg \min_{i=1,\ldots,C} \|\boldsymbol{x} - \boldsymbol{D}_i\hat{\boldsymbol{\alpha}}_i\|_2 \tag{2}$$

where $\hat{\boldsymbol{\alpha}}_i$ contains the coefficients in $\hat{\boldsymbol{\alpha}}$ belonging to ith class.

B. Multiple Pixels' JSRC

Although SRC achieves a good performance, one drawback of SRC is that it can only deal with single test sample. In HSI, neighboring pixels probably belong to the same material so that they usually are strongly correlated with each other. In order to capture such spatial correlations, JSRC [17] assuming that pixels within a small neighborhood can be simultaneously represented using different linear combinations of a few common atoms from a dictionary. Specifically, the size of a region centered at test pixel $\boldsymbol{x}_t$ is denoted as L, and pixels within such a region are denoted by $\{\boldsymbol{x}_i\}, i = 1, \ldots, L$. These pixels can also be stacked into a matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t, \ldots, \boldsymbol{x}_L] \in \Re^{b \times L}$. Each pixel $\boldsymbol{x}_i$ can be approximated using a linear combination of the certain training samples. Then, it can be compactly represented as

$$\begin{aligned} \boldsymbol{X} &= [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t, \ldots, \boldsymbol{x}_L] = [\boldsymbol{D}\boldsymbol{\alpha}_1, \ldots, \boldsymbol{D}\boldsymbol{\alpha}_t, \ldots, \boldsymbol{D}\boldsymbol{\alpha}_L] \\ &= \boldsymbol{D}[\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_t, \ldots, \boldsymbol{\alpha}_L] = \boldsymbol{D}\boldsymbol{A} \end{aligned} \tag{3}$$

where $\boldsymbol{D} \in \Re^{b \times N}$ is the dictionary, $\boldsymbol{A} = [\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_t, \ldots, \boldsymbol{\alpha}_L] \in \Re^{N \times L}$ is the sparse coefficient matrix corresponding to $\boldsymbol{X}$. Because neighboring pixels $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i, \ldots, \boldsymbol{x}_L]$ are linearly approximated by a small set of common atoms, the positions of nonzero coefficients should be in a few nonzero rows of the sparse coefficient matrix $\boldsymbol{A}$. It can be expressed as the following optimization problem:

$$\hat{\boldsymbol{A}} = \arg\min_{A} \|\boldsymbol{X} - \boldsymbol{D}\boldsymbol{A}\|_F \quad \text{s.t.} \quad \|\boldsymbol{A}\|_{\text{row},0} \leq K \tag{4}$$

where $\|A\|_{\text{row},0}$ denotes the joint sparse constraint, which enforces to select a number of the most representative nonzero rows in $\boldsymbol{A}$; $\|\bullet\|_F$ is the Frobenius norm, which is used to calculate the reconstruction error. After $\hat{\boldsymbol{A}}$ is recovered, the label of test pixel $\boldsymbol{x}_t$ can be decided by the minimal total error, i.e.,

$$\text{label}(\boldsymbol{x}_t) = \arg \min_{i=1,\ldots,C} r_i(\boldsymbol{X}) = \arg \min_{i=1,\ldots,C} \|\boldsymbol{X} - \boldsymbol{D}_i\hat{\boldsymbol{A}}_i\|_2 \tag{5}$$

where $\hat{\boldsymbol{A}}_i$ denotes the rows in $\hat{\boldsymbol{A}}$ associated with the ith class.

Compared with the pixelwise SRC model, in terms of accuracy, JSRC incorporating spatial information of local regions can deliver much better classification results.

III. Proposed Method

Even though JSRC achieves a good performance, incorporating multiple types of features will improve the classification performance further. In addition, the requirement of all the coefficients sharing the same sparsity pattern is too strict. In this paper, we propose a class-level joint sparse representation model with multifeature learning for HSI classification. Furthermore, a kernel class-level joint sparse representation model is used to handle nonlinearity in the data.

A. Class-Level Joint Sparse Representation Model

In the multifeature cases, we assume that there are S different types of discriminative features to describe each pixel. For each

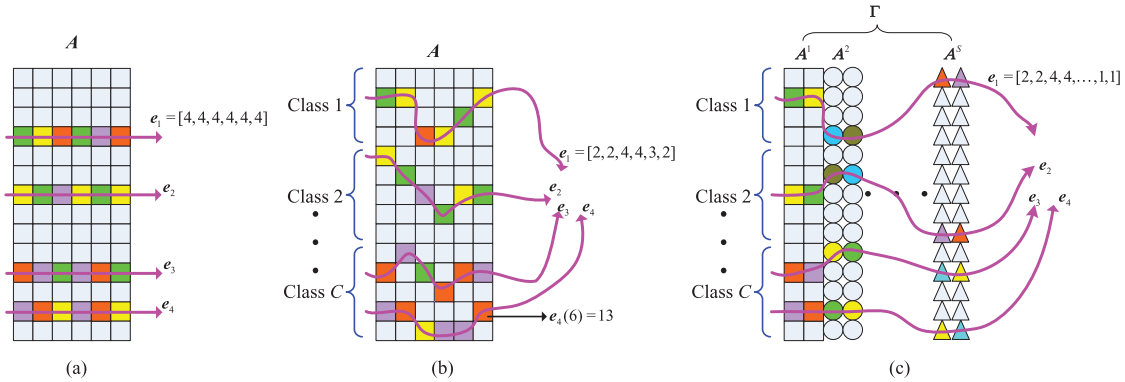


Fig. 2. Illustration of different sparsity patterns of coefficients: (a) row sparsity pattern; (b) class-level sparsity pattern; and (c) class-level sparsity pattern for multiple inputs' and multiple dictionaries' case.

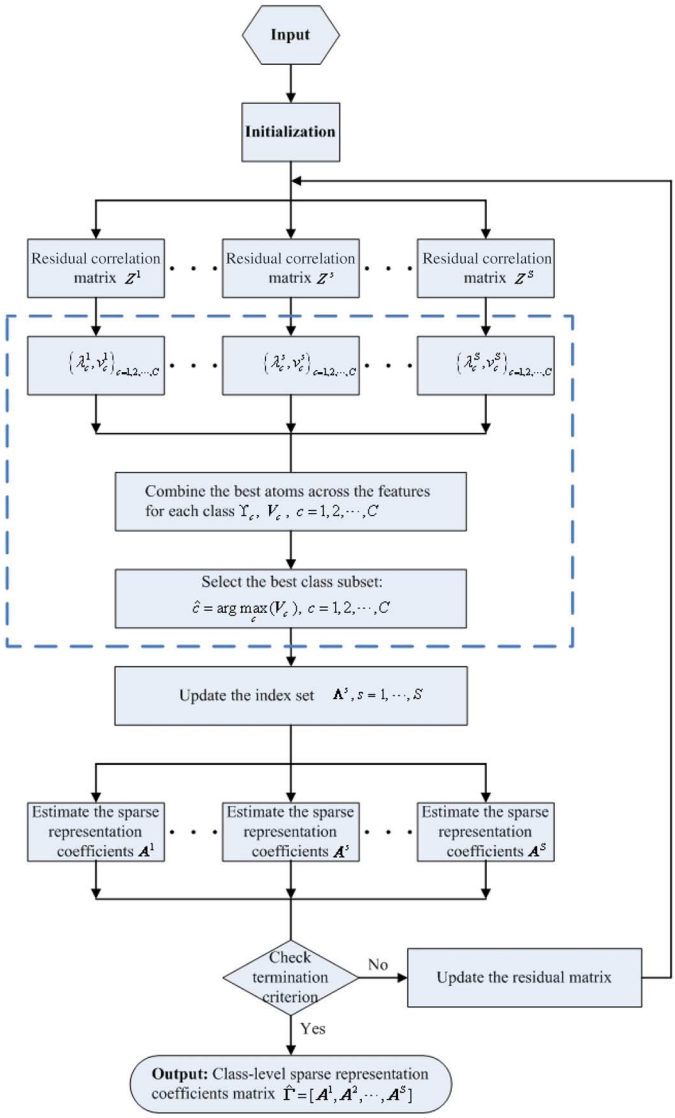


Fig. 3. Flowchart of the optimization algorithm.

pixel $\boldsymbol{x}$, we denote $\boldsymbol{x}^s \in \Re^{b^s}(s=1,\ldots,S)$ as the sth type of feature vector. $\boldsymbol{D}^s=[\boldsymbol{D}_1^s,\ldots,\boldsymbol{D}_c^s,\ldots,\boldsymbol{D}_C^s]$ is the dictionary of the sth type of feature ($\boldsymbol{D}_c^s \in \Re^{b^s \times N_c}$ is the cth class subdictionary; b^s is the dimension of the sth feature vector). In the case of only the spectral feature, the JSRC (described in Section II-B) takes the spatial information into consideration. Similarly, in the multifeature case, such spatial correlations can also be used for each kind of feature. Given a test sample $\boldsymbol{x}_t$, we get a matrix $\boldsymbol{X}=[\boldsymbol{x}_1,\ldots,\boldsymbol{x}_t,\ldots,\boldsymbol{x}_L]$ of size L via simultaneously stacking all the pixels in the neighborhood centered at the hyperspectral pixel $\boldsymbol{x}_t$. For S different types of features, in the same way, we can construct the matrix set $\{\boldsymbol{X}^s\}_{s=1,\cdots,S}=\{[\boldsymbol{x}_1^s,\ldots,\boldsymbol{x}_t^s,\ldots,\boldsymbol{x}_L^s]\}_{s=1,\ldots,S}$ which contains S matrices sized $b^s \times L$ for each neighborhood patch. Based on the framework of JSRC, we can get joint sparse representation coefficient matrices $\{\boldsymbol{A}^s\}_{s=1,\ldots,S} \in \Re^{N\times L}$ associated with the corresponding feature dictionary $\{\boldsymbol{D}^s\}_{s=1,\ldots,S}$.

For one neighborhood pixels $\boldsymbol{X}$, multiple different types of features $\{\boldsymbol{X}^s\}_{s=1,\ldots,S}$ are extracted from the different perspectives. Since the same training samples from different features construct corresponding subdictionaries $\{\boldsymbol{D}^s\}_{s=1,\ldots,S}$, respectively, it is reasonable that these features may share some similarities. Therefore, it can be assumed that the representation coefficients $\{\boldsymbol{A}^s\}_{s=1,\ldots,S}$ over their associated subdictionaries should share same sparsity pattern among all the features, i.e., the selected atoms tend to be the same. However, due to the difference of feature space, the representation coefficients need not to be identical. Thus, the desired sparsity pattern among all the features should be at class level, i.e., the representation coefficients corresponding to different feature spaces have freedom to select their own appropriate atoms within each class.

According to [22], [41], a support vector $\boldsymbol{e}_k \in \Re^L$ is defined as the kth index set of the nonzero coefficients in matrix $\boldsymbol{A} \in \Re^{N\times L}$. Support vector $\boldsymbol{e}_k$ includes one and only one index

TABLE I
PARAMETERS FOR FEATURES

Datasets	Features	Parameters	No. of dimension
Indian Pines/ Pavia University/ Pavia Center images	GLCM	Base image: PC1, PC2, PC3, PC4; Measure: angular second moment, contrast, entropy, variance, correlation; Window: 3,7,11; Direction: averaging the extracted features over four directions.	60
	DMP	Base image: PC1, PC2 ; Size of structuring elements: 3,5,7,9 ; Morphological operators: opening and closing.	16
	3D-WT	Base image :PC1, PC2, PC3, PC4; Level : 2.	60

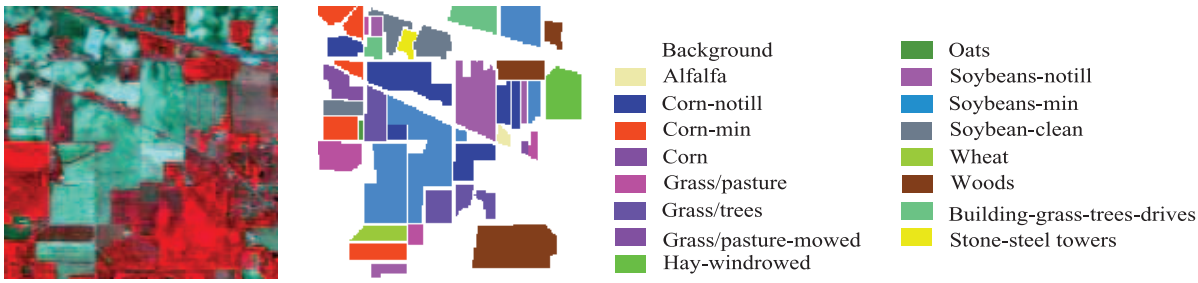


Fig. 4. Indian Pines image. (a) Three-band color composite image. (b) Reference image.

TABLE II
SIXTEEN REFERENCE CLASSES IN THE INDIAN PINES IMAGE

Class		Samples	
No	Name	Train	Test
1	Alfalfa	3	51
2	Corn-notill	72	1362
3	Corn-min	42	792
4	Corn	12	222
5	Grass/pasture	25	472
6	Grass/trees	38	709
7	Grass/pasture–mowed	2	24
8	Hay-windrowed	5	464
9	Oats	1	19
10	Soybeans-notill	49	919
11	Soybeans-min	123	2345
12	Soybean-clean	31	583
13	Wheat	11	201
14	Woods	65	1229
15	Building-grass-trees-drives	19	361
16	Stone-steel towers	5	90
Total		523	9843

for each column of matrix $\boldsymbol{A}$, where $\boldsymbol{e}_k(j)$ is for the jth column of $\boldsymbol{A}$. For example, $\boldsymbol{e}_1 = [2, 2, 4, 4, 3, 2]$, and $\boldsymbol{e}_4(6) = 13$ is shown in Fig. 2(b). $\boldsymbol{a}_{e_k} \in \Re^L$ is denoted as the coefficient vector associated with the support vector $\boldsymbol{e}_k$

$$\begin{aligned}\boldsymbol{a}_{e_k} &= \boldsymbol{A}(\boldsymbol{e}_k) \\ &= [\boldsymbol{A}(\boldsymbol{e}_k(1), 1), \boldsymbol{A}(\boldsymbol{e}_k(2), 2), \ldots, \boldsymbol{A}(\boldsymbol{e}_k(L), L)]^T. \quad (6)\end{aligned}$$

$\|\cdot\|_{\text{row},0}$-norm denotes the number of nonzero rows of a coefficient matrix. For row sparsity pattern, the entries in the support vector $\boldsymbol{e}_k$ are identical, e.g., $\boldsymbol{e}_1 = [4, 4, 4, 4, 4, 4]$ as shown in Fig. 2(a). Thus, $\boldsymbol{a}_{e_k} = \boldsymbol{A}(e_k) = \boldsymbol{A}(\varepsilon_k, :)$, where ε_k is a value of support vector $\boldsymbol{e}_k$. $\|\mathbf{A}\|_{\text{row},0}$ can be described as

$$\|\boldsymbol{A}\|_{\text{row},0} = \|[\|\boldsymbol{A}(\varepsilon_1, :)\|_2, \|\boldsymbol{A}(\varepsilon_2, :)\|_2, \ldots]\|_0. \quad (7)$$

For class-level sparsity pattern, the entries in the support vector $\boldsymbol{e}_k$ need not to be identical, but still belong to the same class. For instance, $\boldsymbol{e}_1 = [2, 2, 4, 4, 3, 2]$, as shown in Fig. 2(b), is a support vector whose entries are different but belong to Class 1. We can get a vector $\boldsymbol{a}_{\boldsymbol{e}_k}$ of nonzero coefficients according to (6). Then, $\|\boldsymbol{A}\|_{\text{class},0}$ can be described as

$$\|\boldsymbol{A}\|_{\text{class},0} = \|[\|\boldsymbol{a}_{\boldsymbol{e}_1}\|_2, \|\boldsymbol{a}_{\boldsymbol{e}_2}\|_2, \ldots]\|_0. \quad (8)$$

In this paper, we use the class-level sparsity pattern with multiple dictionaries. Suppose S dictionaries, we enforce the columns of coefficient matrix $\boldsymbol{A}^s$ have the same nonzero positions, i.e., $\|\boldsymbol{A}^s\|_{\text{row},0}$. The entries in the support vector $\boldsymbol{e}_k^s$ are identical, e.g., $\boldsymbol{e}_1^1 = [2, 2]$, $\boldsymbol{e}_1^2 = [4, 4]$, $\boldsymbol{e}_1^S = [1, 1]$ as shown in Fig. 2(c), while $\boldsymbol{e}_k^s$ could be different with each other. Following class-level sparsity pattern definition, we define $\|\boldsymbol{\Gamma}\|_{\text{class},0}$ for multiple dictionaries

$$\|\boldsymbol{\Gamma}\|_{\text{class},0} = \|[\|\boldsymbol{a}_{\boldsymbol{E}_1}\|_2, \|\boldsymbol{a}_{\boldsymbol{E}_2}\|_2, \ldots]\|_0 \quad (9)$$

where $\boldsymbol{\Gamma} = \left[\boldsymbol{A}^1, \boldsymbol{A}^2, \ldots, \boldsymbol{A}^S\right]$. $\boldsymbol{E}_k$ is a index vector formed by the collection of $\boldsymbol{e}_k^s (s = 1, \ldots, S)$, i.e., $\boldsymbol{E}_k = [\boldsymbol{e}_k^1, \boldsymbol{e}_k^2, \ldots, \boldsymbol{e}_k^S]$, and $\boldsymbol{a}_{\boldsymbol{E}_k} = \boldsymbol{\Gamma}(\boldsymbol{E}_k) = [\boldsymbol{\Gamma}(\boldsymbol{E}_k(1), 1), \boldsymbol{\Gamma}(\boldsymbol{E}_k(2), 2), \ldots]^T$.

Under this assumption, class-level joint sparse representation model combining multiple features (CL-JSRC) can be formulated as

$$\hat{\boldsymbol{\Gamma}} = \arg\min_{\boldsymbol{\Gamma}} \sum_{s=1}^{S} \|\boldsymbol{X}^s - \boldsymbol{D}^s \boldsymbol{A}^s\|_F \quad \text{s.t. } \|\boldsymbol{\Gamma}\|_{\text{class},0} \le K \quad (10)$$

where $\boldsymbol{\Gamma} = \left[\boldsymbol{A}^1, \boldsymbol{A}^2, \ldots, \boldsymbol{A}^S\right]$ is a matrix formed by concatenating the coefficient matrices. The objective function aims at minimizing the reconstruction error for all features. The constraint term $\|\boldsymbol{\Gamma}\|_{\text{class},0}$ includes two-level sparse penalties. On the one hand, the columns within each type of feature share the same sparsity pattern, i.e., the positions of nonzero coefficients lie in the same row, which is called as joint sparse constraint (norm) $\ell_{row,0}$. On the other hand, the positions of nonzero coefficients from different features are allowed to be in different rows of $\boldsymbol{\Gamma}$ but still belong to the same class. This means that pixels from different features have freedom to select different atoms within each class. In this way, it not only preserves the spatial information by joint sparse constraint but also utilizes additional complementary information from different features by class-level sparse constraint.

Once $\hat{\boldsymbol{\Gamma}}$ is obtained, we can determine the label of the center pixel $\boldsymbol{x}_t$ through calculating the total residual errors between $\boldsymbol{X}^s$ and the approximations obtained over their corresponding subdictionaries $\{\boldsymbol{D}_i^s\}_{\substack{s=1,\ldots,S \\ i=1,\ldots,C}}$. The center pixel $\boldsymbol{x}_t$ belongs to the class that yields the minimal total residual as

$$\begin{aligned}\text{label}(\boldsymbol{x}_t) &= \arg\min_{i=1,\ldots,C} r_i(\boldsymbol{X}^s) \\ &= \arg\min_{i=1,\ldots,C} \sum_{s=1}^{S} \|\boldsymbol{X}^s - \boldsymbol{D}_i^s \boldsymbol{A}_i^s\|_F \quad (11)\end{aligned}$$

TABLE III
CLASSIFICATION ACCURACY (%) USING SINGLE FEATURE FOR THE INDIAN PINES IMAGE ON THE TEST SET

Methods	SVM				SRC				JSRC			
Features	Spectral	GLCM	DMP	3D-WT	Spectral	GLCM	DMP	3D-WT	Spectral	GLCM	DMP	3D-WT
1	64.71	57.84	**82.75**	20.00	59.61	58.04	**83.14**	50.00	**90.98**	85.49	85.10	87.45
2	79.65	80.40	**83.48**	70.54	59.55	74.40	**87.85**	68.66	88.11	87.48	**90.92**	88.51
3	67.07	68.14	**87.83**	49.09	57.15	61.15	**89.18**	55.77	83.69	79.77	**86.74**	80.81
4	54.91	52.16	**91.35**	24.10	38.02	48.69	**88.92**	63.78	82.21	75.05	**87.34**	82.03
5	91.06	86.27	**92.22**	59.96	85.93	83.60	**93.41**	85.55	**95.04**	91.04	91.36	90.64
6	95.73	94.43	**96.11**	58.69	93.47	92.81	**94.36**	70.35	96.16	**96.46**	92.98	89.04
7	71.67	60.00	**92.50**	19.17	77.08	67.92	**97.08**	76.25	**83.33**	78.33	81.67	82.92
8	**97.56**	94.94	97.16	83.74	97.11	94.40	**97.13**	96.72	**99.91**	98.75	95.54	99.46
9	39.47	44.74	**51.58**	8.95	41.05	50.00	56.32	**66.32**	42.11	**62.11**	48.95	59.47
10	70.11	**74.32**	71.64	74.11	70.07	72.91	**83.48**	77.49	**91.16**	88.08	86.83	89.26
11	83.93	82.58	**90.22**	88.46	72.11	80.54	**90.51**	78.25	91.28	94.36	**96.17**	91.56
12	**74.39**	51.87	73.46	35.68	48.13	47.80	**78.78**	46.76	**83.88**	74.31	79.78	73.12
13	**98.51**	93.28	97.61	70.60	**97.96**	94.33	97.91	93.28	93.03	89.50	**98.61**	92.94
14	96.28	94.11	**97.99**	87.42	93.30	94.24	**98.19**	88.57	98.79	97.53	**98.90**	94.04
15	47.09	63.41	**94.93**	19.64	36.95	54.49	**96.45**	58.17	75.43	84.68	**88.53**	79.00
16	**84.89**	39.33	78.11	16.11	85.67	**89.33**	79.56	86.44	85.11	**93.00**	74.67	92.89
OA	81.59 ±0.89	79.76 ±1.29	**88.32** ±2.18	68.80 ±3.17	72.06 ±0.88	77.35 ±0.71	**90.52** ±0.36	74.26 ±0.51	90.74 ±0.48	90.01 ±1.10	**91.88** ±0.96	88.68 ±1.09
AA	76.06 ±2.91	71.11 ±3.50	**86.18** ±2.37	49.14 ±5.30	69.60 ±1.85	72.82 ±1.82	**88.27** ±1.34	72.75 ±2.20	86.39 ±1.68	86.00 ±2.81	**86.50** ±2.74	85.82 ±2.25
Kappa	78.93 ±1.08	76.82 ±1.51	**86.65** ±2.52	62.93 ±4.11	68.11 ±1.00	74.09 ±0.79	**89.21** ±0.41	70.58 ±0.59	89.43 ±0.55	88.57 ±1.26	**90.71** ±1.11	87.08 ±1.25

where $\boldsymbol{A}_i^s$ is the subset of the coefficient matrix $\boldsymbol{A}^s$ associated with class *i*. The outline of the proposed CL-JSRC framework is illustrated in Fig. 1.

The above-mentioned CL-JSRC method is proposed for the linear representation with feature vectors. If the features are encoded as similarity or kernel matrices, or the classes in the feature space are not linearly separable, then the kernel extension of CL-JSRC (KCL-JSRC) has significance in combining multiple-feature kernels. The aim of kernel extension is to make the classes become linearly separable in a higher dimensional space by using a kernel function ϕ to map the data from the original feature space to reproducing kernel Hilbert space [42].

Considering the general case of S different types of features, $\boldsymbol{X}^s (s = 1, \ldots, S)$ in kernel space can be written as $\Phi(\boldsymbol{X}^s) = [\phi(\boldsymbol{x}_1^s), \phi(\boldsymbol{x}_2^s), \ldots, \phi(\boldsymbol{x}_L^s)]$. Similarly, the dictionary corresponding to *s*th type of feature can be represented in kernel space as $\Phi(\boldsymbol{D}^s)$. In the new space, (10) can be written as

$$\hat{\boldsymbol{\Gamma}} = \arg\min_{\Gamma} \sum_{s=1}^{S} \|\Phi(\boldsymbol{X}^s) - \Phi(\boldsymbol{D}^s)\boldsymbol{A}^s\|_F \quad \text{s.t.} \quad \|\boldsymbol{\Gamma}\|_{\text{class},0} \leq K \tag{12}$$

where $\boldsymbol{\Gamma} = \left[\boldsymbol{A}^1, \boldsymbol{A}^2, \ldots, \boldsymbol{A}^S\right]$. In kernel space, the information from all kinds of features is integrated via the coefficient matrix $\boldsymbol{\Gamma}$ which is enforced the class-level sparse restraint. We define $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle$ for some given kernel function κ. This can be reformulated in terms of kernel matrices as

$$\hat{\boldsymbol{\Gamma}} = \arg\min_{\Gamma} \sum_{s=1}^{S} \left(\text{trace}\left(\boldsymbol{A}^{s^T} \boldsymbol{K}_{\boldsymbol{D}^s, \boldsymbol{D}^s} \boldsymbol{A}^s \right) - 2\text{trace}\left(\boldsymbol{A}^{s^T} \boldsymbol{K}_{\boldsymbol{D}^s, \boldsymbol{X}^s} \right) \right) \quad \text{s.t.} \quad \|\boldsymbol{\Gamma}\|_{\text{class},0} \leq K \tag{13}$$

where $\boldsymbol{K}_{\boldsymbol{D}^s, \boldsymbol{D}^s}$ is the kernel matrix whose (*i*, *j*)th entry is $\kappa(\boldsymbol{d}_i^s, \boldsymbol{d}_j^s)$, and $\boldsymbol{K}_{\boldsymbol{D}^s, \boldsymbol{X}^s}$ is the matrix whose (*i*, *j*)th entry is $\kappa(\boldsymbol{d}_i^s, \boldsymbol{x}_j^s)$. After obtaining $\hat{\boldsymbol{\Gamma}}$, classification can be done by assigning the class label as

$$\text{label}(\boldsymbol{x}) = \arg\min_{i=1,\cdots,C} \sum_{s=1}^{S} \left(\text{trace}\left(\boldsymbol{K}_{\boldsymbol{X}^s, \boldsymbol{X}^s} \right) - 2\text{trace}\left(\boldsymbol{A}_i^{s^T} \boldsymbol{K}_{\boldsymbol{D}_i^s, \boldsymbol{X}^s} \right) + \text{trace}\left(\boldsymbol{A}_i^{s^T} \boldsymbol{K}_{\boldsymbol{D}_i^s, \boldsymbol{D}_i^s} \boldsymbol{A}_i^s \right) \right) \tag{14}$$

where $\boldsymbol{D}_i^s$ is the subdictionary associated with the *i*th class and $\boldsymbol{A}_i^s$ is the subset of the coefficient matrix $\boldsymbol{A}^s$ associated with class *i*.

B. Optimization Algorithm

For the aforementioned problems, generally we could use approximation algorithms [22], [43], such as SOMP algorithm [22], [44] to solve them. In this paper, MSOMP algorithm is used to approximately solve this problem. SOMP is commonly used to solve the simultaneous sparse approximation problem as (4). It can obtain a simultaneous sparse approximation of multiple inputs using the same atoms from a certain dictionary. While the problem in this paper as (10) requests a simultaneous sparse approximation of multiple inputs over multiple dictionaries (features), we cannot use SOMP algorithm to solve them directly. In particular, problem in (4) is essential in (10) when $S = 1$. In essence, MSOMP has the same theoretical basis with SOMP. The algorithm structure of MSOMP is similar to that of the SOMP, including the following general steps: 1) compute the current residual correlation matrix; 2) find new atom that best approximates the current residue; 3) update the index set of atoms; 4) calculate the sparse representation coefficients based on new atoms set and update the residual matrix; and 5) check termination criterions. The modification or the major difference [22] between MSOMP and SOMP lies in the greedy selection rule in Step 2), which is detailed as follows.

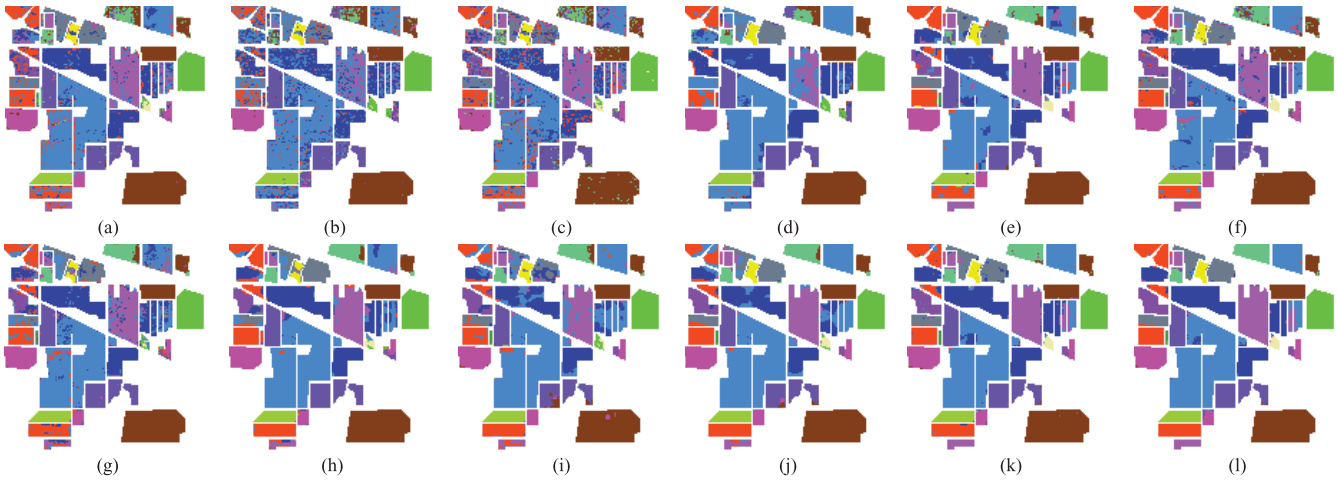


Fig. 5. Indian Pines image: the classification maps obtained by (a) SVM (OA=82.01); (b) CRC (OA=66.31); (c) SRC (OA=72.22); (d) JCRC (OA=76.38); (e) JSRC (OA=90.98); (f) SVMCK (OA=90.57); (g) CRC-MTL (OA=84.05); (h) JCRC-MTL (OA=90.03); (i) SRC-MTL (OA=84.69); (j) JSRC-MTL (OA=91.59); (k) CL-JSRC (OA=95.54); and (l) KCL-JSRC (OA=97.25).

TABLE IV
CLASSIFICATION ACCURACY (%) FOR THE INDIAN PINES IMAGE ON THE TEST SET

Methods	Single feature methods				Multifeature methods						
	CRC	JCRC	SRC	JSRC	SVMCK	CRC-MTL	JCRC-MTL	SRC-MTL	JSRC-MTL	CL-JSRC	KCL-JSRC
1	0	0	59.61	90.98	68.82	18.43	36.47	33.14	69.41	**92.16**	88.63
2	69.54	86.63	59.55	88.11	91.12	81.48	85.35	65.21	76.78	**95.32**	95.21
3	29.63	41.02	57.15	83.69	87.90	78.38	87.31	77.89	85.71	91.77	**96.81**
4	2.07	0.41	38.02	82.21	81.71	59.46	76.13	76.22	91.35	93.51	**97.39**
5	70.32	68.88	85.93	95.04	94.41	90.28	92.33	94.28	**96.97**	94.05	96.17
6	94.72	**99.72**	93.47	96.16	96.85	97.38	98.46	89.35	95.67	97.93	99.51
7	0	0	77.08	83.33	72.50	39.17	46.25	**94.17**	89.58	89.17	90.42
8	99.78	**100**	97.11	99.91	97.11	99.87	99.98	99.89	99.98	99.96	99.98
9	0	0	41.05	42.11	60.00	31.05	12.63	**80.00**	52.63	66.32	61.58
10	29.08	26.70	70.07	91.16	86.41	67.23	77.23	71.04	85.54	94.65	**95.02**
11	76.80	92.81	72.11	91.28	91.76	87.40	92.75	94.58	**98.47**	97.00	98.06
12	44.29	66.19	48.13	83.88	84.22	64.56	73.79	42.23	67.60	84.51	**89.85**
13	98.11	99.75	97.96	93.03	94.93	99.05	98.76	**99.85**	**99.85**	98.51	**99.85**
14	97.09	99.47	93.30	98.79	96.44	98.35	98.80	99.30	99.32	**99.67**	99.59
15	36.40	54.29	36.95	75.43	69.81	81.44	91.02	80.08	92.85	92.88	**94.79**
16	74.78	88.00	85.67	85.11	93.00	84.22	87.89	97.56	**98.44**	93.56	93.56
OA	66.73 ±0.76	76.27 ±0.95	72.06 ±0.88	90.74 ±0.48	90.56 ±0.75	84.21 ±0.53	89.28 ±1.07	83.16 ±1.17	90.71 ±1.16	95.45 ±0.72	**96.81** ±0.71
AA	51.41 ±0.68	57.74 ±0.44	69.60 ±1.85	86.26 ±1.68	85.44 ±2.41	73.61 ±1.38	78.45 ±2.04	80.92 ±1.95	87.51 ±2.24	92.56 ±2.65	**93.53** ±2.35
Kappa	61.24 ±0.88	72.27 ±1.12	68.11 ±1.00	89.43 ±0.55	89.24 ±0.86	81.87 ±0.62	87.73 ±1.24	80.56 ±1.39	89.33 ±1.35	94.81 ±0.83	**96.36** ±0.81

In SOMP, the process of finding the index of the atom that best approximates all residuals from one type of feature can be expressed as

$$\lambda_{iter} = \arg \max_{i=1,,2,\ldots,N} \|(\boldsymbol{R}_{iter-1})^T \boldsymbol{d}_i\|_p \tag{15}$$

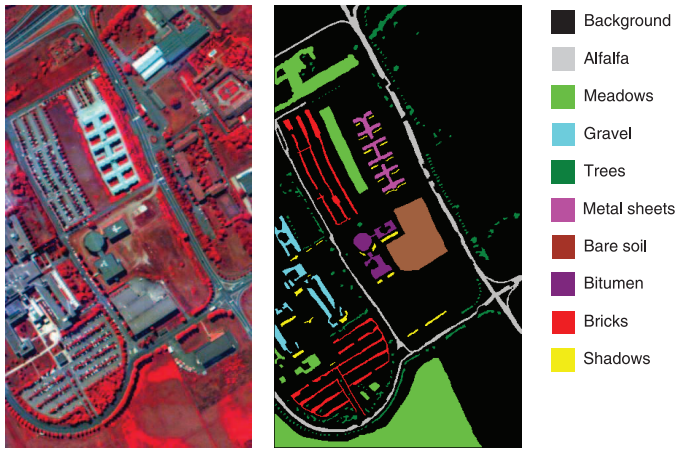
where $p \geq 1$, and $\boldsymbol{d}_i$ is ith atom of $\boldsymbol{D}$. In particular, when $p = 1$, its maximum has the equivalent expressions

$$\max_{i=1,,2,\ldots,N} \|(\boldsymbol{R}_{iter-1})^T \boldsymbol{d}_i\|_1 = \|(\boldsymbol{R}_{iter-1})^T \boldsymbol{D}\|_{1,1} = \|\boldsymbol{Z}\|_{1,1} \tag{16}$$

where $\boldsymbol{Z} = (\boldsymbol{R}_{iter-1})^T \boldsymbol{D}$ and $\|\boldsymbol{Z}\|_{1,1}$is the maximum ℓ_1-norm of any column of $\boldsymbol{Z}$. Thus, the atoms maximizing the sum of absolute correlations are selected in SOMP algorithm.

In MSOMP algorithm, the support of the solution is sequentially updated as SOMP (i.e., the atoms in each dictionary $\boldsymbol{D}^s$ are sequentially selected). But the selected atom needs to simultaneously yield the best approximation to all of the residuals over multiple-feature dictionaries. On the other hand, the atom selection criteria of MSOMP are much different from the one of SOMP as shown in the dashed box in Fig. 3. After computing the current residual correlation matrix for each feature $\boldsymbol{Z}^s = (\boldsymbol{R}^s)^T \boldsymbol{D}^s$, the new candidates are selected via the following steps.

Step 1) Find the best representation atom for cth class and sth type of feature: $\{\lambda_c^s, v_c^s\} = \arg \max_{i,z} \|\boldsymbol{Z}^s(:, i)\|_p$, where i belong to cth class atoms' indexes, i.e., the



(a) (b)

Fig. 6. University of Pavia image. (a) Three-band color composite image. (b) Reference image.

TABLE V
NINE REFERENCE CLASSES IN THE UNIVERSITY OF PAVIA IMAGE

Class		Samples	
No	Name	Train	Test
1	Asphalt	66	6565
2	Meadows	186	18 463
3	Gravel	20	2079
4	Trees	30	3034
5	Metal sheets	13	1332
6	Bare soil	50	4979
7	Bitumen	13	1317
8	Bricks	37	3645
9	Shadows	10	937
Total		425	42351

selected atom's indexes λ_c^s correspond to the largest ℓ_p-norm of columns belong to cth class and v_c^s correspond to the value. According to [22], small values of p prefer atoms which dedicate to numerous different inputs at once, without regard to the contribution magnitude. When $0 \leq p < 1$, it is nonconvex case which may make the problem more complicated. Larger values of p trend to atoms which make a lot of contributions to someone. It is better to set $p = 1$ when the selected atoms can offer the closest approximation to each column of the test matrix. If columns of the test matrix can be more successfully approximated with a collection of different atoms, the value of p should be set to a larger one. We set $p \geq 1$ as the literature [17].

Step 2) Combine the best atoms across the features into a support $\Upsilon_c = \lambda_c^1 \cup \lambda_c^2 \cdots \cup \lambda_c^S, \quad c = 1, 2, \ldots, C$ and calculate the total of coefficients $\boldsymbol{V}_c = \sum_{s=1}^{S} v_c^s$ for each class.

Step 3) Choose the best class subset $\hat{c} = \arg\max_c(\boldsymbol{V}_c)$, $c = 1, 2, \ldots, C$ and update the class-level support set $\boldsymbol{\Lambda}_{iter} = \boldsymbol{\Lambda}_{iter-1} \cup \Upsilon_{\hat{c}}$ defined as the index set of the selected atoms.

The process of the selected atoms is different from SOMP. Thus, we called it MSOMP. After obtaining the support set, we estimate the sparse representation coefficients $\boldsymbol{A}^s$

$$\boldsymbol{A}^s = \left(\left(\boldsymbol{D}^s_{\boldsymbol{\Lambda}^s_{iter}}\right)^T \boldsymbol{D}^s_{\boldsymbol{\Lambda}^s_{iter}}\right)^{-1} \left(\boldsymbol{D}^s_{\boldsymbol{\Lambda}^s_{iter}}\right)^T \boldsymbol{X}^s, \quad s = 1, \ldots, S \tag{17}$$

where $\boldsymbol{\Lambda}^s_{iter}$ is the subset of $\boldsymbol{\Lambda}_{iter}$ associated with the $s$thfeature. It will stop and output joint sparse representation coefficients $\hat{\boldsymbol{\Gamma}}$ if the termination criteria are fulfilled, otherwise update the residual matrix $\boldsymbol{R}^s = \boldsymbol{X}^s - \boldsymbol{D}^s_{\boldsymbol{\Lambda}^s_{iter}} \boldsymbol{A}^s$ and return to Step 1).

Similar to the linear method, we propose a kernelized MSOMP method (denoted as KMSOMP) to efficiently deal with (12) or (13). In KMSOMP, the correlation between a dictionary atom $\phi(\boldsymbol{d}_i^s)$ and a pixel $\phi(\boldsymbol{x}^s)$ is computed by the dot product $\kappa(\boldsymbol{d}_i^s, \boldsymbol{x}^s) = \langle \phi(\boldsymbol{d}_i^s), \phi(\boldsymbol{x}^s) \rangle$. In the same way, the correlation between $\boldsymbol{D}^s$ and $\boldsymbol{X}^s$ is denoted as $\mathbf{K}_{\boldsymbol{D}^s, \boldsymbol{X}^s}$. At each iteration, the orthogonal projection coefficients of $\Phi(\boldsymbol{X}^s)$ onto the selected atoms set $\{\phi(\boldsymbol{d}_i^s)\}_{i \in \boldsymbol{\Lambda}^s_{iter}}$ are computed by

$$\boldsymbol{A}^s = \left((\boldsymbol{K}_{\boldsymbol{D}^s, \boldsymbol{D}^s})_{\boldsymbol{\Lambda}^s_{iter}, \boldsymbol{\Lambda}^s_{iter}}\right)^{-1} (\boldsymbol{K}_{\boldsymbol{D}^s, \boldsymbol{X}^s})_{\boldsymbol{\Lambda}^s_{iter}, :} \tag{18}$$

where $\boldsymbol{\Lambda}^s_{iter}$ is the selected atoms subset of $\boldsymbol{\Lambda}_{iter}$ associated with the sth feature. In order to have a stable inversion, we add a regularization term $\mu \boldsymbol{I}$ when computing the projection $\boldsymbol{A}^s$

$$\boldsymbol{A}^s = \left((\boldsymbol{K}_{\boldsymbol{D}^s, \boldsymbol{D}^s})_{\boldsymbol{\Lambda}^s_{iter}, \boldsymbol{\Lambda}^s_{iter}} + \mu \boldsymbol{I}\right)^{-1} (\boldsymbol{K}_{\boldsymbol{D}^s, \boldsymbol{X}^s})_{\boldsymbol{\Lambda}^s_{iter}, :} \tag{19}$$

where $\boldsymbol{I}$ is an identity matrix whose dimensionality is derived from the context, μ is a small scalar and is set as $\mu = 10^{-5}$ in our implementation. Since the matrix is usually invertible and regularization is dispensable, this parameter has little influence on classification performance.

Based on the selected atoms set $\{\phi(\boldsymbol{d}_i^s)\}_{i \in \boldsymbol{\Lambda}^s_{iter}} = \Phi(\boldsymbol{D}^s)_{:, \boldsymbol{\Lambda}^s_{iter}}$, the residual matrix between $\Phi(\boldsymbol{X}^s)$ and its approximation is then expressed as

$$\begin{aligned} \Phi(\boldsymbol{R}^s) &= \Phi(\boldsymbol{X}^s) - \Phi(\boldsymbol{D}^s)_{:, \boldsymbol{\Lambda}^s_{iter}} \\ &\quad \times \left((\boldsymbol{K}_{\boldsymbol{D}^s, \boldsymbol{D}^s})_{\boldsymbol{\Lambda}^s_{iter}, \boldsymbol{\Lambda}^s_{iter}} + \mu \boldsymbol{I}\right)^{-1} (\boldsymbol{K}_{\boldsymbol{D}^s, \boldsymbol{X}^s})_{\boldsymbol{\Lambda}^s_{iter}, :} \\ &= \Phi(\boldsymbol{X}^s) - \Phi(\boldsymbol{D}^s)_{:, \boldsymbol{\Lambda}^s_{iter}} \boldsymbol{A}^s. \end{aligned} \tag{20}$$

It should be noted that the residual matrix $\Phi(\boldsymbol{R}^s)$ in (20) cannot be calculated directly, while we can compute the correlation between $\Phi(\boldsymbol{R}^s)$ and dictionary $\Phi(\boldsymbol{D}^s)$ as follows:

$$\begin{aligned} \boldsymbol{Z}^s &= \langle \Phi(\boldsymbol{D}^s), \Phi(\boldsymbol{R}^s) \rangle = \boldsymbol{K}_{\boldsymbol{D}^s, \boldsymbol{X}^s} - (\boldsymbol{K}_{\boldsymbol{D}^s, \boldsymbol{D}^s})_{:, \boldsymbol{\Lambda}^s_{iter}} \\ &\quad \times \left((\boldsymbol{K}_{\boldsymbol{D}^s, \boldsymbol{D}^s})_{\boldsymbol{\Lambda}^s_{iter}, \boldsymbol{\Lambda}^s_{iter}} + \mu \boldsymbol{I}\right)^{-1} (\boldsymbol{K}_{\boldsymbol{D}^s, \boldsymbol{X}^s})_{\boldsymbol{\Lambda}^s_{iter}, :} \\ &= \boldsymbol{K}_{\boldsymbol{D}^s, \boldsymbol{X}^s} - (\boldsymbol{K}_{\boldsymbol{D}^s, \boldsymbol{D}^s})_{:, \boldsymbol{\Lambda}^s_{iter}} \boldsymbol{A}^s. \end{aligned} \tag{21}$$

C. Computational Complexity Analysis

Suppose that multifeature test matrix is $\left\{\boldsymbol{X}^s \in \Re^{b^s \times L}\right\}_{s=1,\ldots,S}$, structural dictionary is $\left\{\boldsymbol{D}^s \in \Re^{b^s \times N}\right\}_{s=1,\ldots,S}$, and sparsity degree is K. Much more computational load is inevitably required for multifeature learning and extended contextual information, especially for large-scale dictionary cases. The main cost of the sparse

TABLE VI
CLASSIFICATION ACCURACY (%) USING SINGLE FEATURE FOR THE UNIVERSITY OF PAVIA IMAGE ON THE TEST SET

Methods	SVM				SRC				JSRC			
Features	Spectral	GLCM	DMP	3D-WT	Spectral	GLCM	DMP	3D-WT	Spectral	GLCM	DMP	3D-WT
1	83.90	91.01	**93.77**	86.26	71.20	85.03	84.41	**87.66**	66.40	88.66	87.95	**91.13**
2	94.75	96.47	**97.35**	92.22	92.05	93.73	**97.09**	85.84	95.74	95.99	**97.89**	89.66
3	63.58	41.43	**65.04**	58.09	53.67	40.77	**56.76**	51.90	61.86	46.90	61.90	**66.36**
4	86.11	93.37	**93.70**	85.13	76.61	90.43	**90.64**	81.85	80.22	88.53	**93.75**	88.61
5	98.36	**99.56**	72.91	92.00	99.43	**99.66**	83.90	98.84	**99.95**	99.70	89.90	99.68
6	62.63	58.14	**81.84**	58.35	42.85	63.43	**64.12**	62.32	47.76	69.59	**71.66**	67.88
7	73.37	50.99	65.28	**74.41**	70.78	45.90	75.05	**75.76**	81.94	54.17	77.43	**89.29**
8	76.92	83.99	**89.35**	81.17	73.23	**82.08**	72.21	80.40	80.97	85.98	79.21	**87.70**
9	78.39	**98.01**	69.03	91.68	88.18	94.73	84.02	**96.77**	86.47	95.66	89.21	**99.84**
OA	84.69 ±1.89	85.83 ±0.88	**90.04** ±2.13	83.60 ±1.66	77.91 ±0.65	83.70 ±0.67	**85.28** ±0.69	81.27 ±0.63	81.00 ±0.71	86.75 ±0.64	**88.51** ±0.68	86.47 ±0.42
AA	79.78 ±5.51	79.21 ±2.22	**80.92** ±7.75	79.92 ±3.26	74.22 ±0.78	77.31 ±0.74	78.69 ±1.62	**80.15** ±0.83	77.92 ±0.98	80.57 ±0.81	83.21 ±1.24	**86.68** ±0.70
Kappa	79.43 ±2.68	80.86 ±1.26	**86.70** ±2.87	77.66 ±2.58	70.21 ±0.88	78.16 ±0.88	**80.28** ±0.94	75.13 ±0.83	74.36 ±1.00	82.22 ±0.84	**84.65** ±0.93	82.01 ±0.55

TABLE VII
CLASSIFICATION ACCURACY (%) FOR THE UNIVERSITY OF PAVIA IMAGE ON THE TEST SET

Methods	Single feature methods				Multifeature methods						
	CRC	JCRC	SRC	JSRC	SVMCK	CRC-MTL	JCRC-MTL	SRC-MTL	JSRC-MTL	CL-JSRC	KCL-JSRC
1	73.64	95.01	71.20	66.40	93.06	93.71	95.42	95.98	96.82	96.70	**97.79**
2	94.45	97.31	92.05	95.74	97.83	97.27	98.88	99.34	99.76	98.63	**99.82**
3	28.33	30.71	53.67	61.86	78.26	66.95	71.06	83.16	86.22	**86.23**	83.93
4	82.19	81.52	76.61	80.22	91.66	96.27	97.08	**99.02**	98.28	96.64	97.44
5	99.85	99.99	99.43	99.95	99.50	99.82	99.86	**100**	**100**	**100**	**100**
6	26.12	21.57	42.85	47.76	83.41	75.91	81.10	56.34	64.59	77.29	**87.31**
7	3.45	0.08	70.78	81.94	92.40	84.47	86.56	88.66	**93.04**	92.01	88.48
8	42.58	51.50	73.23	80.97	87.78	93.01	95.13	97.20	95.96	97.85	**98.38**
9	51.55	85.67	88.18	86.47	99.05	78.27	91.57	99.77	**99.83**	97.64	96.23
OA	70.99 ±0.80	76.51 ±1.23	77.91 ±0.65	81.00 ±0.71	93.04 ±0.60	91.54 ±0.53	93.92 ±0.56	92.46 ±0.45	93.87 ±0.36	94.82 ±0.30	**96.52** ±0.41
AA	55.79 ±1.49	62.60 ±1.90	74.22 ±0.78	77.92 ±0.98	91.44 ±0.60	87.30 ±1.38	90.74 ±1.42	91.05 ±1.21	92.72 ±0.87	93.76 ±0.84	**94.71** ±0.66
Kappa	59.88 ±1.18	67.23 ±1.76	70.21 ±0.88	74.36 ±1.00	90.72 ±0.81	88.68 ±0.71	91.87 ±0.75	89.80 ±0.69	91.73 ±0.50	93.07 ±0.41	**95.35** ±0.55

recovery algorithm is the inversion of matrix. In MSOMP algorithm, only one atom is added to the support set at each iteration; thus, the maximal size of the matrix to be inversed is $K \times K$. Therefore, the complexity of MSOMP is $\boldsymbol{O}\left(\sum_{s=1}^{S} KNLb^s\right)$. The KCL-JSRC model needs to compute the kernel matrix $\{\boldsymbol{K}_{\boldsymbol{D}^s,\boldsymbol{D}^s}\}_{s=1,\ldots,S}$ and $\{\boldsymbol{K}_{\boldsymbol{D}^s,\boldsymbol{X}^s}\}_{s=1,\ldots,S}$. The computational complexity will increase correspondingly. Note that the matrix $\{\boldsymbol{K}_{\boldsymbol{D}^s,\boldsymbol{D}^s}\}_{s=1,\ldots,S}$ can be precomputed, and $\{\boldsymbol{K}_{\boldsymbol{D}^s,\boldsymbol{X}^s}\}_{s=1,\ldots,S}$ costs little because the number of neighboring pixels L is usually small. Therefore, the kernel trick does not visibly increase the complexity while outperforming the linear methods, as will be shown in Section IV.

IV. EXPERIMENTS AND PERFORMANCE COMPARISONS

To verify the efficiency and effectiveness of the proposed CL-JSRC and KCL-JSRC, we conducted experiments on three real hyperspectral images: the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) Indian Pines image, the Reflective Optics System Imaging Spectrometer (ROSIS) University of Pavia image, and the ROSIS Center of Pavia image. Detailed comparisons among existing methods (including CR-based and SR-based methods) and the proposed methods are made in this section.

In our experiments, each pixel is described by four different types of features (original spectral value feature, gray-level co-occurrence matrix feature (GLCM) [27], differential morphological profiles (DMP) feature [27] and 3-D wavelet transform feature (3-D-WT) [20]). The parameter values for different kinds of features are set to be identical to the corresponding references and listed in Table I.

The proposed CL-JSRC and KCL-JSRC are compared with the widely used classification methods, including single-feature-based methods (SVM [4], CRC [24], JCRC [23], SRC-Pixel-wise [40], and JSRC [17]) and multifeature-based methods (SVMCK [13], CRC-MTL [25], JCRC-MTL [36], SRC-MTL [33], and JSRC-MTL [32]). The SVM classifier is implemented with radial basis function (RBF) kernels using support vector machines library. SVMCK uses a summation kernel, where the mean of neighborhood pixels is used as the spatial information. In these experiments, the accuracy for each class (CA), overall accuracy (OA), average accuracy (AA), and the kappa coefficient measure (Kappa) are adopted to evaluate the performance of different classifiers. In order to reduce the

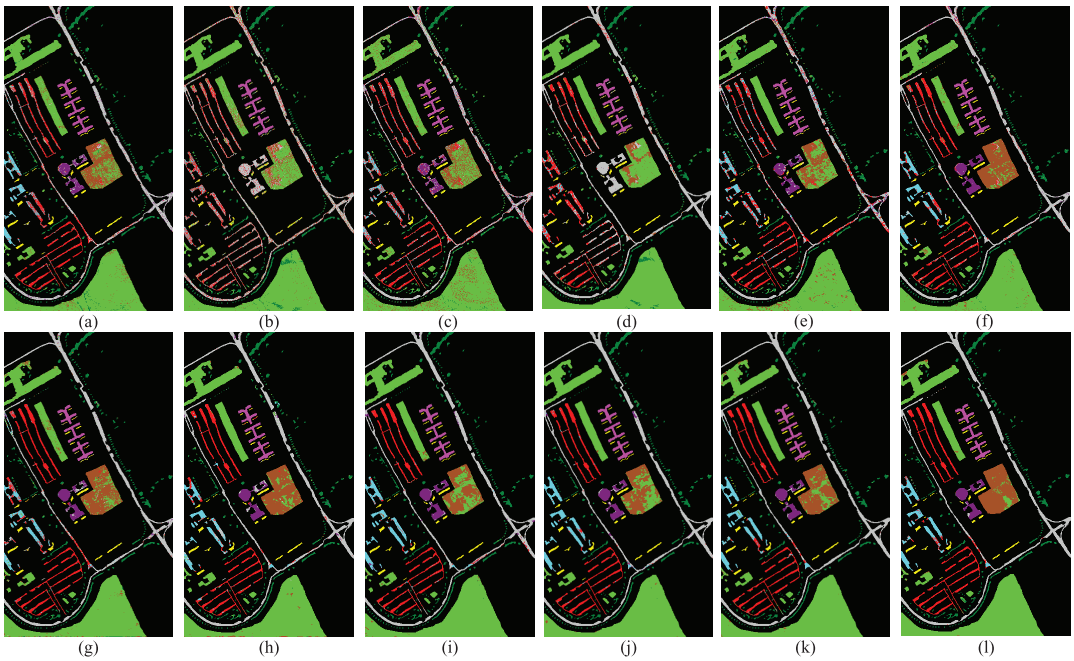


Fig. 7. University of Pavia image: the classification maps obtained by (a) SVM (OA=86.80); (b) CRC (OA=70.96); (c) SRC (OA=77.92); (d) JCRC (OA=76.67); (e) JSRC (OA=81.24); (f) SVMCK (OA=93.01); (g) CRC-MTL (OA=91.63); (h) JCRC-MTL (OA=93.99); (i) SRC-MTL (OA=92.73); (j) JSRC-MTL (OA=93.86); (k) CL-JSRC (OA=94.83); and (l) KCL-JSRC (OA=96.76).

possible bias due to the random samples, every evaluation index takes an average of 10 runs for each case.

A. *Experimental Results of AVIRIS Data Set*

The first HSI in our experiments is the Indian Pines image which is obtained by AVIRIS system. The size of this image is 145×145, and spatial resolution is 20 m per pixel. The image has 220 spectral channels in the wavelength range $[0.2, 2.4]$ μm. In the experiments, we use 200 spectral bands after removing 20 bands because of noise and water absorption phenomena. Fig. 4(b) shows the reference data which is consisted of 16 classes. Most classes are various types of crops. This image is widely used to evaluate the performance of classification algorithms. It is a challenge because of class-imbalanced problem and the mixed pixel existence. In this experiment, around 5% of the labeled pixels for each class are chosen randomly for training (total 523 samples) and the remaining samples (9843 samples) are used to test the classifiers (see Table II).

First, we investigate the complementary properties of the aforementioned multiple features and the necessity of combining multiple features. Table III shows the classification results of SVM, SRC, and JSRC with different types of features, in which the best results for each quality index are labeled in bold. Only a part of classification maps are shown in Fig. 5 because of space constraint. From these classification results in Table III, it is observed that the original spectral value feature-based classifiers and the DMP-feature-based classifiers lead to better performance than the others, whereas each type of feature can achieve the best accuracies on certain classes. For instance, with the JSRC method, the spectral value feature performs the best with Class 1, 5, 7, 8, 10, and 12; the GLCM texture feature achieves the best classification result with Class 6, 9, and 16; the DMP shape feature obtains the best results on the rest of the classes. It can draw similar conclusions with the other classification methods. In other words, different features reflect different aspects of the discriminative information of the HSI. Thus, it is feasible to combine the multiple features to improve the performance of classification.

Next, we analyze the performance of the proposed CL-JSRC and KCL-JSRC with other multiple-feature learning methods. In this part, using the single type of feature, the original spectral value feature, the CR-based methods (CRC and JCRC) and SR-based methods (SRC and JSRC) are, respectively, applied as classifiers. Multifeature-based methods include SVMCK, CR-based algorithms (CRC-MTL and JCRC-MTL), SR-based algorithms (SRC-MTL and JSRC-MTL), and the proposed methods (CL-JSRC and KCL-JSRC). The parameters of SVMCK are obtained by cross-validation. The regularization parameters for CRC, JCRC, CRC-MTL, JCRC-MTL, SRC-MTL, and JSRC-MTL algorithms range from 10^{-6} to 10^{-1}. RBF kernels are used in KCL-JSRC, and parameter γ ranges from 10^{-3} to 10^{3}. The neighborhood size L for JCRC, JSRC, JCRC-MTL, JSRC-MTL, CL-JSRC, and KCL-JSRC is set to 25. The results averaged over 10 runs for forementioned methods are listed in Table IV, and the classification maps are shown in Fig. 5. First, comparing with the single-feature-based methods, all multifeature methods dramatically improve the classification performance in terms of accuracy. In the visual maps shown in Fig. 5, it can be observed that multifeature fusion-based methods can alleviate the "salt-and-pepper" phenomenon and obtain a smoothing map. It proves the rationality and necessity of combining the multiple features. Second, as shown in Table IV, the joint representation methods (JCRC, JSRC, JCRC-MTL, JSRC-MTL) can get better results than those methods without considering neighboring pixels (CRC, SRC, CRC-MTL, and SRC-MTL). It

demonstrates that the spatial prior knowledge is helpful for HSI classification. Finally, the proposed CL-JSRC and KCL-JSRC obtain the best results on most classes, as shown in Table IV. Especially, the proposed algorithms achieve a satisfying classification performance on "Corn" ("Corn-notill," "Corn-min," and "Corn") and "Soybeans" ("Soybeans-notill," "Soybeans-min," and "Soybeans-clean") which are challenging tasks for the traditional methods due to quite similar spectral characteristics. CL-JSRC method has a better performance than other compared approaches in terms of OA, AA, and Kappa coefficients. It can be seen that CL-JSRC has a remarkable improvement over the SVM classifier with the spectral feature and an enhancement of 5% on OA over the SVMCK. Comparing with the CRC-MTL and SRC-MTL, the gains (in OA, AA, and Kappa coefficients) of the CL-JSRC are more than 10%. Moreover, the CL-JSRC has an improvement of 5% on OA over the JCRC-MTL and JSRC-MTL, respectively. It demonstrates that the proposed CL-JSRC successfully integrates multifeature information and contextual neighborhood information in joint sparse representation framework. Furthermore, the KCL-JSRC shows a better performance than CL-JSRC in term of accuracy and classification map, which means the kernel extensions are good solutions for HSI classification. Overall, the proposed algorithms achieve satisfying performance.

B. Experimental Results of ROSIS Urban Data Over Pavia

In this section, two ROSIS urban data sets are used to evaluate the proposed methods. The first data set is the University of Pavia image whose spatial resolution is 1.3 m. It consists of 610×340 pixels which has 103 bands (range from 0.43 to 0.86 μm) after removing 12 noisiest bands. This image covers nine ground-truth classes as shown in Fig. 6(b). Around 1% samples from each class are selected randomly for training, and the remaining samples are used for testing (see Table V).

The classification results of SVM, SRC, and JSRC with each single type of feature are shown in Table VI. In this scene, the parameters of the classifiers are set as same as that of Indian Pines image. The neighborhood size L is set to a small value (3×3) because this image is an urban area without the large homogeneity. As Indian Pines image, similar conclusions can be drawn from the experimental results of the University of Pavia image. It can be seen that each type of feature can achieve the best accuracies on certain classes. That is to say, different features can reflect different aspects of the discriminative information and are complementary to each other.

The classification results of the methods based on a single spectral value feature (CRC, JCRC, SRC, and JSRC) and multifeature-based methods (SVMCK, CRC-MTL, JCRC-MTL, SRC-MTL, JSRC-MTL, CL-JSRC, and KCL-JSRC) are summarized in Table VII. And corresponding classification maps are presented in Fig. 7. As is shown in Table VII, multifeature methods perform much better than single-feature methods in terms of accuracy and yield smoother visual effect. It demonstrates the superiority of the multifeature methods. It can be observed that the OA of the proposed CL-JSRC is slightly higher than those of JCRC-MTL and JSRC-MTL, while KCL-JSRC outperforms the other multiple-feature learning methods in term of accuracy. Comparing with SVMCK, JCRC-MTL, and JSRC-MTL, the gains (in OA, AA, and Kappa coefficients) of the KCL-JSRC are more than 3%, i.e., more than 1000 pixels. And, KCL-JSRC shows a better performance than CL-JSRC. It is proved that the kernel extensions play an important role in HSI classification. In addition, the proposed CL-JSRC and KCL-JSRC can successfully integrate multifeature information and contextual neighborhood information in joint sparse representation framework.

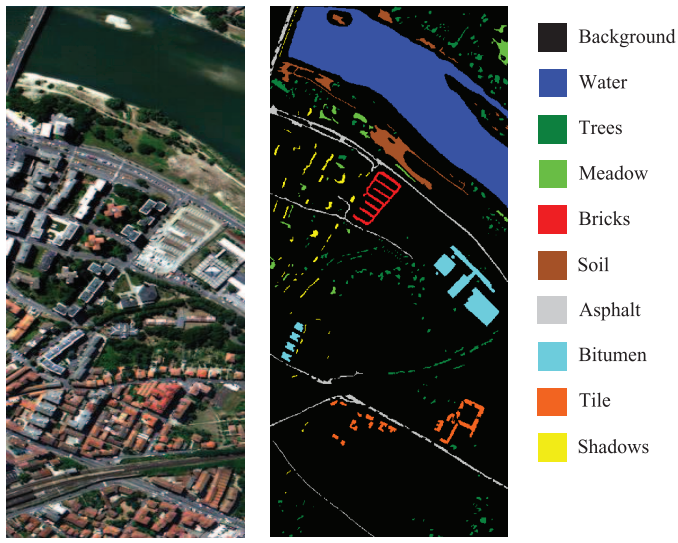


Fig. 8. Center of Pavia image. (a) Three-band color composite image. (b) Reference image.

TABLE VIII
NINE REFERENCE CLASSES IN THE CENTER OF PAVIA IMAGE

Class		Samples	
No	Name	Train	Test
1	Water	15	65263
2	Trees	15	6493
3	Meadow	15	2890
4	Bricks	15	2125
5	Soil	15	6534
6	Asphalt	15	7570
7	Bitumen	15	7272
8	Tile	15	3107
9	Shadows	15	2150
Total		135	103404

The second data set is the Center of Pavia image which has 102 spectral bands after taking away 13 noisy bands. A patch with size of 1096×492 is used in our experiments. Nine ground-truth classes are shown in Fig. 8(b). Some interested objects are scattered across the scene, which increase the difficulty of classification. In our experiments, about 15 samples of each class are randomly selected for training and the remaining samples for testing (see Table VIII). The same experimental setting as the University of Pavia image is adopted.

The classification results of the SVM, SRC, and JSRC with different features are shown in Table IX. Table X shows the classification results of above-mentioned multifeature learning methods. A part of classification maps are shown in Fig. 9. From Table IX, it can be observed that the spectral value feature and 3-D-WT feature-based classifiers achieve better

TABLE IX
CLASSIFICATION ACCURACY (%) USING SINGLE FEATURE FOR THE CENTER OF PAVIA IMAGE ON THE TEST SET

Methods	SVM				SRC				JSRC			
Features	Spectral	GLCM	DMP	3D-WT	Spectral	GLCM	DMP	3D-WT	Spectral	GLCM	DMP	3D-WT
1	94.39	97.98	96.73	**98.01**	**98.99**	98.16	97.46	97.85	**99.49**	99.31	99.37	98.49
2	87.24	78.41	**91.21**	81.90	79.55	70.59	**88.89**	87.54	84.17	74.18	87.42	**87.95**
3	84.01	65.07	83.86	**85.48**	**87.16**	61.39	78.67	86.87	**90.25**	66.59	77.97	87.84
4	79.95	92.73	80.61	**92.92**	75.44	**93.41**	86.74	90.70	87.36	**95.57**	85.23	91.85
5	82.78	81.13	49.73	**83.11**	79.94	77.65	56.99	**82.53**	78.35	79.67	61.82	**83.57**
6	84.46	77.00	65.35	**91.06**	75.41	79.93	64.90	**80.00**	77.89	**83.22**	62.59	81.92
7	80.13	76.51	52.98	**84.64**	81.95	76.85	64.48	**86.16**	86.60	80.17	66.73	**86.77**
8	**97.38**	95.34	84.34	87.94	**96.64**	90.76	82.16	95.30	**99.23**	92.94	82.09	96.84
9	98.41	91.93	**99.88**	96.290	82.35	90.45	**99.09**	95.88	90.59	92.18	**98.19**	96.61
OA	91.06 ±1.48	91.41 ±1.04	87.04 ±1.53	**93.81** ±1.67	92.41 ±0.67	90.79 ±1.00	88.49 ±1.21	**93.54** ±0.89	94.01 ±0.62	92.63 ±0.75	89.83 ±0.89	**94.33** ±0.65
AA	87.64 ±1.18	84.01 ±2.30	78.30 ±2.03	**89.04** ±2.98	84.16 ±1.19	82.14 ±1.57	79.94 ±1.07	**89.20** ±0.86	88.21 ±1.12	85.07 ±1.65	80.16 ±1.12	**90.20** ±0.84
Kappa	85.20 ±2.31	85.41 ±1.73	78.22 ±2.50	**89.49** ±2.76	87.04 ±1.16	84.36 ±1.60	80.57 ±1.93	**89.03** ±1.44	89.74 ±1.07	87.39 ±1.28	82.59 ±1.50	**90.33** ±1.07

TABLE X
CLASSIFICATION ACCURACY (%) FOR THE CENTER OF PAVIA IMAGE ON THE TEST SET

Methods	Single feature methods				Multifeature methods						
	CRC	JCRC	SRC	JSRC	SVMCK	CRC-MTL	JCRC –MTL	SRC-MTL	JSRC-MTL	CL-JSRC	KCL-JSRC
1	93.34	98.68	98.99	99.49	98.60	97.75	98.12	98.37	**99.96**	99.89	99.58
2	83.96	87.43	79.55	84.17	89.76	88.60	90.60	**93.46**	93.22	91.81	90.27
3	86.34	90.94	87.16	90.25	88.01	91.49	92.18	88.99	94.59	94.01	**96.23**
4	43.86	61.51	75.44	87.36	90.02	85.13	94.07	96.61	98.44	97.93	**98.45**
5	61.07	75.83	79.94	78.35	87.44	87.53	**93.30**	91.66	84.11	90.40	91.14
6	48.64	71.87	75.41	77.89	90.73	86.71	89.06	84.96	91.54	93.36	**97.28**
7	76.63	88.12	81.95	86.60	87.62	84.34	89.34	85.24	84.85	89.17	**89.35**
8	92.29	99.62	96.64	99.23	98.82	96.17	98.50	99.17	98.62	99.32	**99.72**
9	53.13	90.37	82.35	90.59	97.15	79.81	96.58	96.84	98.56	**99.60**	99.40
OA	84.18 ±1.11	92.70 ±0.71	92.41 ±0.67	94.01 ±0.62	95.49 ±0.41	93.92 ±0.63	95.81 ±0.49	95.42 ±0.65	96.61 ±0.60	97.32 ±0.52	**97.46** ±0.53
AA	71.03 ±0.97	84.95 ±1.26	84.16 ±1.19	88.21 ±1.12	92.02 ±1.04	88.61 ±0.79	93.56 ±0.73	92.81 ±0.89	93.77 ±0.92	95.07 ±1.03	**95.76** ±1.01
Kappa	73.92 ±1.60	87.56 ±1.18	87.04 ±1.16	89.74 ±1.07	92.32 ±0.69	89.69 ±1.03	92.88 ±0.82	92.27 ±1.01	94.17 ±0.99	95.40 ±0.88	**95.64** ±0.90

performance than the others. Different features reflect different aspects of the discriminative information and are complementary to each other. Thus, it is reasonable to combine the multiple features to improve the performance of classification. It can come to similar conclusions with Indian Pines and Pavia University images. The experimental results show that the proposed CL-JSRC and KCL-JSRC improve about 2% than SVMCK in term of OA and have a gain of 3% over CRC-MTL. They outperform JCRC-MTL and JSRC-MTL with about 1% gain in terms of OA. In this experiment, the improvement of the proposed KCL-JSRC is limited which partly due to CL-JSRC before the kernelization already achieves a very high accuracy of at least 97%, on the other hand, due to kernel transformation may not produce relevant additional information for classification.

C. Running Time and the Effect of the Number of Training Samples

First, we compare the running times of the various classification algorithms including four single-feature algorithms (CRC, JCRC, SRC, and JSRC) and seven multifeature algorithms (SVMCK, CRC-MTL, JCRC-MTL, SRC-MTL, JSRC-MTL, CL-JSRC, and KCL-JSRC). All the programs are executed using MATLAB in the environment of an Intel Core i3-550 CPU 3.20 GHz and RAM 4 GB. Different percentages of samples are randomly chosen to construct a training sample set, and 100 samples are randomly chosen to be test samples. For the Indian Pines image, the running times of the various classifiers with different training sets (from 5% to 40% per class) are shown in Table XI. For the University of Pavia image, we randomly choose 0.25%–4% pixels in each class for training. The running times are shown in Table XII. For the Center of Pavia image, 10–80 labeled pixels in each class are used as the training. The running times for each case are shown in Table XIII.

From Tables XI–XIII, it can be observed that CRC is faster than the other methods but serves the worst classification performance; JCRC and JSRC perform better than CRC and SRC, but they cost more time; JSRC has a better performance than JCRC while costs comparable time. And, it also observed that multiple-feature methods cost more time than those single-feature methods. The main reason is as follows: the computational load is increased because of multitask learning and the extended contextual information [36]. It should be

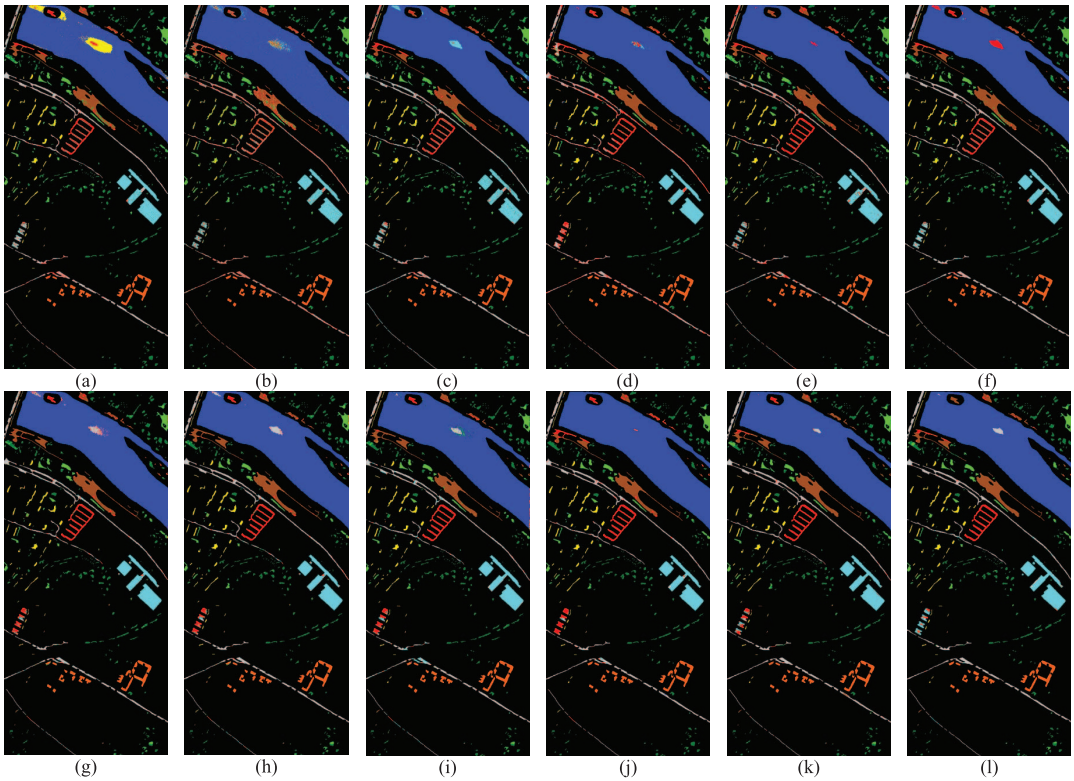


Fig. 9. Center of Pavia image: the classification maps obtained by (a) SVM (OA=90.61); (b) CRC (OA=84.60); (c) SRC (OA=92.51); (d) JCRC (OA=92.58); (e) JSRC (OA=94.01); (f) SVMCK (OA=95.51); (g) CRC-MTL (OA=93.92); (h) JCRC-MTL (OA=95.88); (i) SRC-MTL (OA=95.52); (j) JSRC-MTL (OA=96.80); (k) CL-JSRC (OA=97.30); and (l) KCL-JSRC (OA=97.48).

TABLE XI
RUNNING TIME (S) FOR THE CLASSIFICATION OF THE INDIAN PINES IMAGE

Methods		5%	10%	15%	20%	25%	30%	35%	40%
Single feature methods	CRC	0.12	0.15	0.24	0.43	0.78	1.05	1.24	1.41
	JCRC	0.69	1.24	1.96	2.73	3.50	4.17	4.78	5.58
	SRC	0.23	0.33	0.43	0.63	1.28	1.81	2.18	2.58
	JSRC	1.49	2.31	3.12	4.61	6.27	7.51	8.49	9.54
Multiple feature methods	SVMCK	1.83	3.47	5.04	6.76	8.21	9.95	11.10	13.29
	CRC-MTL	1.19	3.71	8.22	14.29	21.84	31.08	41.79	52.78
	JCRC-MTL	3.73	10.61	20.91	34.28	51.15	72.70	96.15	125.11
	SRC-MTL	17.42	43.70	76.83	120.23	171.49	232.32	301.03	379.39
	JSRC-MTL	45.81	137.04	259.96	422.17	618.56	895.72	1143.95	1503.46
	CL-JSRC	2.98	5.23	7.91	10.39	13.02	15.87	17.78	20.12
	KCL-JSRC	3.04	5.24	7.98	10.49	13.49	16.12	18.38	21.84

noted that the running time of SVMCK is obtained on all the testing pixels (except training pixels). But it has little influence on running time due to the main cost of SVMCK is during the training stage, and SVM is implemented by C++ software to speed up. Compared with SVMCK, these representation-based methods need to execute the algorithms for every test sample. They are time-consuming methods when many test samples are used. However, because there is no training stage, they can be simply extended or changed to predict a new test sample when the training samples are varied. Although SRC-MTL and JSRC-MTL utilize the accelerated proximal gradient method, they also need a few hundred times of iteration to achieve satisfying recognition accuracy [32]. When dealing with a large-scale training sample set as the dictionary, much computing time is required by SRC-MTL and JSRC-MTL. When the projection matrices in CRC-MTL and JCRC-MTL algorithms are computed offline, CRC-MTL and JCRC-MTL cost comparative times with the proposed CL-JSRC and KCK-JSRC. Especially, dealing with a large-scale training sample set as the dictionary, CRC-MTL and JCRC-MTL including matrix inverse operation consume more computing time and memory. Although the optimization problems in the proposed CL-JSRC and KCL-JSRC are NP-hard problems, they can be approximately solved by greedy pursuit algorithms. In this paper, we adopted MSOMP algorithm which can avoid large matrix inverse operation. Thus, it can be seen that the proposed CL-JSRC and KCL-JSRC have advantages in the computational efficiency.

In order to a fair comparison, we make an analysis of computational complexity of some representation-based methods, such as JCRC-MTL [25], [36], JSRC-MTL [32], and the proposed methods. As the analysis of computational complexity described in Section III-C, CL-JSRC

TABLE XII
RUNNING TIME (S) FOR THE CLASSIFICATION OF THE UNIVERSITY OF PAVIA IMAGE

Methods		0.25%	0.5%	1%	2%	3%	4%
Single feature methods	CRC	0.03	0.04	0.05	0.06	0.08	0.10
	JCRC	0.07	0.08	0.11	0.18	0.25	0.32
	SRC	0.12	0.14	0.16	0.22	0.28	0.33
	JSRC	0.27	0.31	0.46	0.52	0.63	0.74
Multiple feature methods	SVMCK	1.19	2.10	3.84	7.91	12.01	28.73
	CRC-MTL	0.12	0.18	0.69	2.56	5.28	8.99
	JCRC-MTL	0.24	0.41	1.24	4.24	8.50	14.80
	SRC-MTL	6.43	11.23	25.92	66.07	114.85	174.59
	JSRC-MTL	7.91	16.00	40.84	122.78	233.39	361.84
	CL-JSRC	0.92	0.94	1.16	1.56	2.01	2.72
	KCL-JSRC	0.98	0.96	1.25	2.05	3.22	4.76

TABLE XIII
RUNNING TIME (S) FOR THE CLASSIFICATION OF THE CENTER OF PAVIA IMAGE

Methods		10	15	20	30	40	60	80
Single feature methods	CRC	0.02	0.03	0.03	0.04	0.04	0.05	0.06
	JCRC	0.05	0.06	0.07	0.08	0.09	0.13	0.16
	SRC	0.11	0.12	0.13	0.15	0.16	0.18	0.20
	JSRC	0.27	0.30	0.31	0.34	0.38	0.41	0.46
Multiple feature methods	SVMCK	2.43	3.36	4.31	6.63	8.51	14.88	73.13
	CRC-MTL	0.12	0.14	0.16	0.31	0.47	1.11	1.88
	JCRC-MTL	0.20	0.25	0.31	0.55	0.85	1.89	3.14
	SRC-MTL	5.04	7.17	9.40	13.86	20.64	35.69	53.56
	JSRC-MTL	7.06	9.64	13.17	20.55	32.47	57.73	89.04
	CL-JSRC	0.79	0.85	0.88	0.98	1.12	1.31	1.48
	KCL-JSRC	0.86	0.92	0.96	1.28	1.39	1.53	1.72

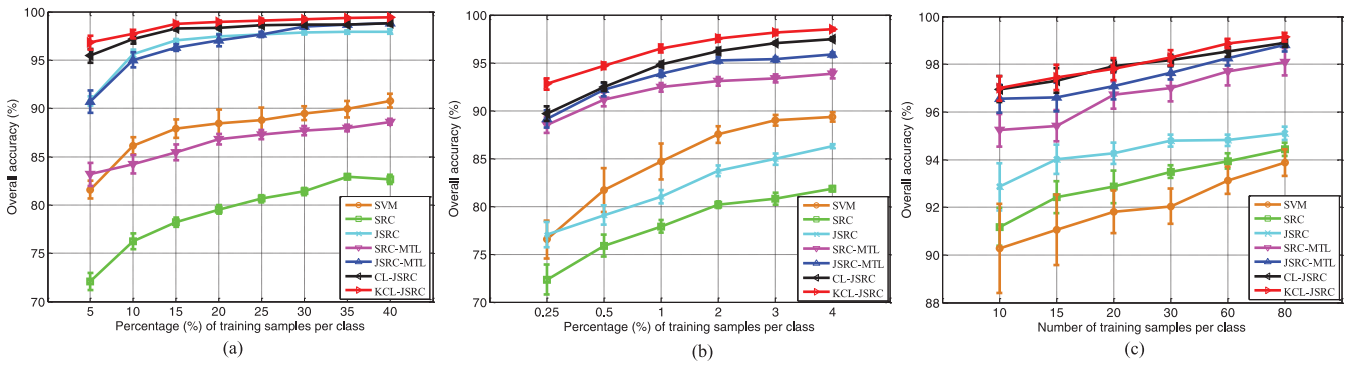


Fig. 10. Effects of the number of training samples on the SVM, SRC, JSRC, SRC-MTL, JSRC-MTL, and the proposed CL-JSRC and KCL-JSRC for (a) Indian Pines; (b) University of Pavia; and (c) Center of Pavia image.

needs $\boldsymbol{O}\left(\sum_{s=1}^{S} KNLb^s\right)$ for every test sample. JCRC-MTL needs $\boldsymbol{O}\left(3SN^2L + NL\sum_{s=1}^{S} b^s\right)$ after the projection matrices are computed offline, and JSRC-MTL needs $\boldsymbol{O}\left(\sum_{s=1}^{S}\left(NLb^s + 2TNLb^s\right)\right)$. Let $B = \sum_{s=1}^{S} b^s$, and the time complexity ratio between CL-JSRC and JCRC-MTL is $\boldsymbol{O}\left(\frac{KNL\sum_{s=1}^{S} b^s}{3SN^2L+NL\sum_{s=1}^{S} b^s}\right) = \boldsymbol{O}\left(\frac{KB}{3SN+B}\right)$. The time complexity ratio between CL-JSRC and JSRC-MTL is $\boldsymbol{O}\left(\frac{\sum_{s=1}^{S} KNLb^s}{\sum_{s=1}^{S}\left(NLb^s+2TNLb^s\right)}\right) \approx \boldsymbol{O}\left(\frac{K}{2T}\right)$. It can be seen that the time complexity ratio is mainly determined by N and B, i.e., the mumble of training sample and the sum of feature dimensions, as sparsity degree K and the number of neighboring pixels L are much smaller than N. When N and B are small, the proposed method has comparative complexity with JCRC-MTL and JCRC-MTL; when N is large, the proposed method has advantages on the complexity. Although the complexity of JSRC-MTL has the same order with the one of CL-JSRC, it needs a few hundred times of iteration to obtain satisfying recognition accuracy, as shown in [32]. In our experiments, the sparsity level K is set about 5, which is much smaller than the times of iteration T. Thus, relatively speaking, CL-JSRC performs faster while exhibits satisfactory classification accuracy.

Finally, the effects of different numbers of training samples to the proposed methods (CL-JSRC and KCL-JSRC) are examined on several real data sets as shown in Fig. 10. SVM, SRC, JSRC, SRC-MTL, and JSRC-MTL are used as baseline

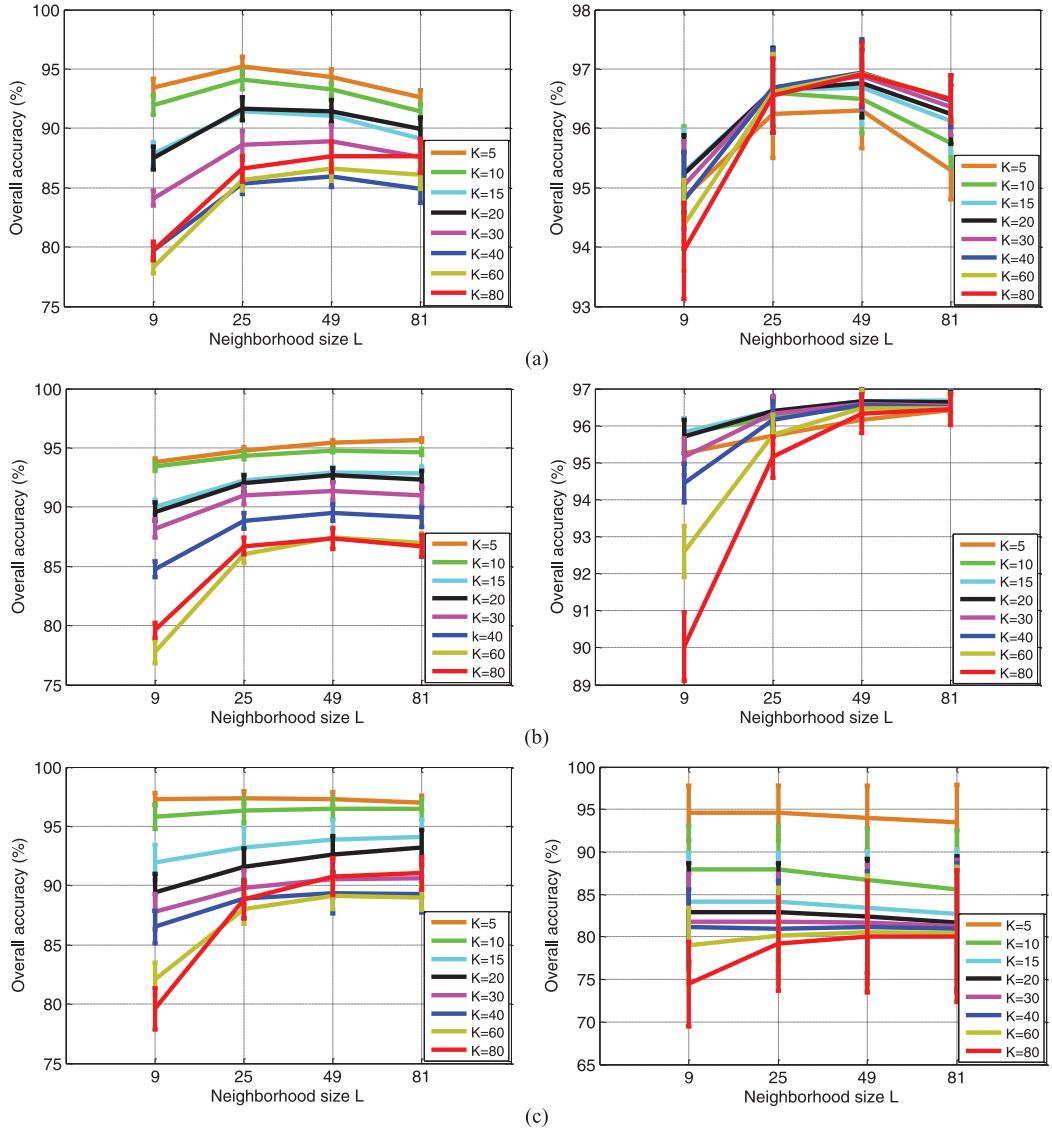


Fig. 11. Effect of the neighborhood size L and the sparsity level K on the performance of CL-JSRC (on the left) and KCL-JSRC (on the right) for (a) Indian Pines; (b) University of Pavia; and (c) Center of Pavia image.

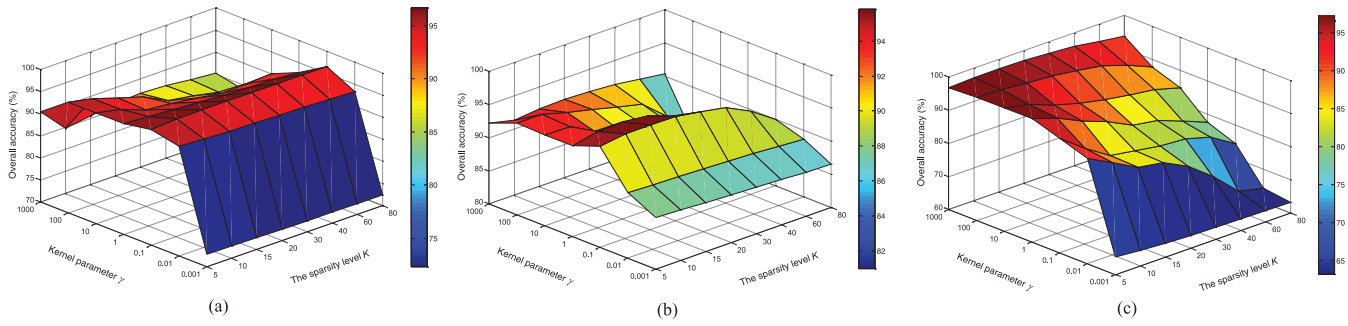


Fig. 12. Effects of RBF kernel parameter γ and sparsity level K using KCL-JSRC on (a) Indian Pines; (b) University of Pavia; and (c) Center of Pavia image.

methods. Training sample set is constructed of samples randomly chosen in different percentages (from 5% to 40% per class for the Indian Pines, from 0.25% to 4% per class for the University of Pavia, from 10 to 80 per class for the Center of Pavia image), and the test set consists of the remaining samples. The results are averaged over ten runs. As is shown in Fig. 10, the performances of all the classifiers generally improve with the number of training samples increases. It is also shown in Fig. 10 that the proposed CL-JSRC and KCL-JSRC provide state-of-the-art results with the limitation in the number of training samples which further corroborate the feasibility of combining multiple features' information. In conclusion, the

proposed CL-JSRC and KCL-JSRC indeed outperform other approaches on all the training samples.

D. Effects of the Parameters

First, we investigate the effect of the neighborhood size L and the sparsity level K on the performance of CL-JSRC and KCL-JSRC. For the Indian Pines image, around 5% of the labeled pixels for each class are chosen randomly for training, and the remaining pixels are used to test the classifiers. For the University of Pavia image, about 1% pixels from each class are selected randomly for training, and the remaining pixels are used for testing. For the Center of Pavia data, 15 pixels from each class compose the training set, and the rest pixels are used as the testing set. In each test, CL-JSRC and KCL-JSRC are executed with different neighborhood sizes L and the sparsity level K. The neighborhood size L ranges from a window 3×3 ($L = 9$) to 9×9 ($L = 81$) and the sparsity level K ranges from 5 to 80. The OA with the associated standard deviations for three testing data set are illustrated in Fig. 11, where the *x*-axis is the neighborhood size L, and the *y*-axis denotes the OA (%). Here, the accuracies are averaged over ten runs for each case, and the bar indicates the standard deviations. Experimental results show that CL-JSRC with small sparsity level K has better performance. As sparsity level K increases, the sparsity of the solution decreases and more atoms from multiple subdictionaries are chosen, which may weaken the discrimination resulting in some performance degradation. On the Indian Pines and University of Pavia data sets, KCL-JSRC favor larger K from 10 to 30. On the University of Pavia image, KCL-JSRC achieves the best accuracies with small K due to few training samples are available in each class. For ROSIS urban data over Pavia, the classification performance has no obvious change as the neighborhood size L increases from 25 to 81. It shows that the proposed methods can successfully utilize contextual neighborhood knowledge. For Indian Pines image, the proposed methods achieve the best performance with $L = 25$ or $L = 49$. If the neighborhood size L is too large, the classification accuracies are significantly reduced. This is because that too many neighboring pixels cannot obtain an effective approximation with few training samples; on the other hand, large neighborhood may bring into neighboring classes.

Next, we examine the effect of RBF parameter γ and sparsity level K on the performance of KCL-JSRC. For three testing data, the RBF parameter γ varies from 10^{-3} to 10^{3}, and the sparsity level K ranges from 5 to 80. The OA results averaged over ten independent runs are shown in Fig. 12, where *x*-axis is the sparsity level K, *y*-axis is the RBF parameter γ, and *z*-axis denotes the OA (%). We can observe from Fig. 12 that KCL-JSRC with $\gamma = 0.1$ can achieve satisfactory classification accuracies at all sparsity levels. For a fixed γ, K prefers a small value ($K = 5$) in consideration of classification accuracy and computational efficiency. From the observation, we can find that the OA is relatively stable with a large range (0.1–10) of γ and a proper K (5–20). The "optimal" γ and K are almost certainly task dependent. In order to obtain the best performance, we should better employ empirical parameters γ and K for different data sets.

V. Conclusion

Recently, some models are developed to deal with hyperspectral image classification through sparse coding theory. These models directly use the single original spectral value feature or multiple types of features with much time consuming. This paper has presented a novel joint sparse representation classification method with class-level sparse constraint. The proposed model not only preserves the spatial information by joint sparse constraint but also utilizes additional complementary information from different features. Thus, the proposed method fuses multiple features' information and spatial information into a more preferable representation. Furthermore, kernel extensions are proposed for nonlinear cases. In this paper, an efficient algorithm based on SOMP is proposed to solve the optimization problems. On real hyperspectral images, the extensive experimental results confirm the efficiency and effectiveness of the proposed CL-JSRC and KCL-JSRC algorithm.

Despite the satisfying performance, the proposed CL-JSRC algorithm could still be further improved in certain aspects. For instance, eliminating effect of noisy and trivial information, considering semantic relationships among the features, adaptively selecting kernel function for each kind of feature, dealing with multisource feature space (spectral, optical, and radar features), etc.

References

[1] E. Christophe, D. Leger, and C. Mailhes, "Quality criteria benchmark for hyperspectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 9, pp. 2103–2114, Sep. 2005.

[2] P. K. Goel, S. O. Prasher, R. M. Patel, J. A. Landry, R. B. Bonnell, and A. A. Viau, "Classification of hyperspectral data by decision trees and artificial neural networks to identify weed stress and nitrogen status of corn," *Comput. Electron. Agric.*, vol. 39, no. 2, pp. 67–93, May 2003.

[3] 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.

[4] 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.

[5] L. Bruzzone, C. Mingmin, and M. Marconcini, "A novel transductive SVM for semisupervised classification of remote-sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 11, pp. 3363–3373, Nov. 2006.

[6] M. Chi and L. Bruzzone, "Semisupervised classification of hyperspectral images by SVMs optimized in the primal," *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 6, pp. 1870–1880, Jun. 2007.

[7] J. Li, J. M. Bioucas-Dias, and A. Plaza, "Semisupervised hyperspectral image classification using soft sparse multinomial logistic regression," *IEEE Geosci. Remote Sens. Lett.*, vol. 10, no. 2, pp. 318–322, Mar. 2013.

[8] F. Ratle, G. Camps-Valls, and J. Weston, "Semisupervised neural networks for efficient hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 5, pp. 2271–2282, May 2010.

[9] Y. Zhong and L. Zhang, "An adaptive artificial immune network for supervised classification of multi-/hyperspectral remote sensing imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 3, pp. 894–909, Mar. 2012.

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

[11] Y. Tarabalka, J. A. Benediktsson, and J. Chanussot, "Spectral-spatial classification of hyperspectral imagery based on partitional clustering techniques," *IEEE Trans. Geosci. Remote Sens.*, vol. 47, no. 8, pp. 2973–2987, Aug. 2009.

[12] Y. Tarabalka, J. A. Benediktsson, J. Chanussot, and J. C. Tilton, "Multiple spectral-spatial classification approach for hyperspectral data," *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 11, pp. 4122–4132, Nov. 2010.

[13] G. Camps-Valls, L. Gomez-Chova, J. Muñoz-Marí, J. Vila-Francés, and J. Calpe-Maravilla, "Composite kernels for hyperspectral image classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 3, no. 1, pp. 93–97, Jan. 2006.

[14] J. Li, P. Reddy Marpu, A. Plaza, J. M. Bioucas-Dias, and J. Atli Benediktsson, "Generalized composite kernel framework for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 51, no. 9, pp. 4816–4829, Sep. 2013.

[15] J. Li, J. M. Bioucas-Dias, and A. Plaza, "Hyperspectral image segmentation using a new bayesian approach with active learning," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 10, pp. 3947–3960, Oct. 2011.

[16] J. Li, J. M. 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, Mar. 2012.

[17] Y. Chen, N. M. Nasrabadi, and T. D. Tran, "Hyperspectral image classification using dictionary-based sparse representation," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 10, pp. 3973–3985, Oct. 2011.

[18] Y. Chen, N. M. Nasrabadi, and T. D. Tran, "Hyperspectral image classification via kernel sparse representation," *IEEE Trans. Geosci. Remote Sens.*, vol. 51, no. 1, pp. 217–231, Jan. 2013.

[19] U. Srinivas, Y. Chen, V. Monga, N. M. Nasrabadi, and T. D. Tran, "Exploiting sparsity in hyperspectral image classification via graphical models," *IEEE Geosci. Remote Sens. Lett.*, vol. 10, no. 3, pp. 505–509, May 2013.

[20] Y. Qian, M. Ye, and J. Zhou, "Hyperspectral image classification based on structured sparse logistic regression and three-dimensional wavelet texture features," *IEEE Trans. Geosci. Remote Sens.*, vol. 51, no. 4, pp. 2276–2291, Apr. 2013.

[21] Z. H. Xue, J. Li, L. Cheng, and P. J. Du, "Spectral-spatial classification of hyperspectral data via morphological component analysis-based image separation," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 1, pp. 70–84, Jan. 2015.

[22] J. A. Tropp, A. C. Gilbert, and M. J. Strauss, "Algorithms for simultaneous sparse approximation. Part I: Greedy pursuit," *J. Signal Process.*, vol. 86, no. 3, pp. 572–588, 2006.

[23] J. Li, H. Zhang, Y. Huang, and L. Zhang, "Hyperspectral image classification by nonlocal joint collaborative representation with a locally adaptive dictionary," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 6, pp. 3707–3719, Jun. 2014.

[24] L. Zhang, M. Yang, X. Feng, Y. Ma, and D. Zhang, "Collaborative representation based classification for face recognition," 2012, arXiv: 1204.2358.

[25] Y. Meng, L. Zhang, D. Zhang, and S. Wang, "Relaxed collaborative representation for pattern classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 2224–2231.

[26] L. Zhang, L. Zhang, D. Tao, and X. Huang, "On combining multiple features for hyperspectral remote sensing image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 3, pp. 879–893, Mar. 2012.

[27] X. Huang and L. Zhang, "An SVM ensemble approach combining spectral, structural, and semantic features for the classification of high-resolution remotely sensed imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 51, no. 1, pp. 257–272, Jan. 2013.

[28] J. Li *et al.*, "Multiple feature learning for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 3, pp. 1592–1606, Mar. 2015.

[29] R. Caruana, "Multi-task learning," *Mach. Learn.*, vol. 28, no. 1, pp. 430–445, 1997.

[30] T. Kato, H. Kashima, M. Sugiyama, and K. Asai, "Multi-task learning via conic programming," in *Advances in Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, 2008.

[31] S. Ozawa, A. Roy, and D. Roussinov, "A multitask learning model for online pattern recognition," *IEEE Trans. Neural Netw.*, vol. 20, no. 3, pp. 430–445, Mar. 2009.

[32] X.-T. Yuan, X. Liu, and S. Yan, "Visual classification with multitask joint sparse representation," *IEEE Trans. Image Process.*, vol. 21, no. 10, pp. 4349–4360, Oct. 2012.

[33] X. Zheng, X. Sun, K. Fu, and H. Wang, "Automatic annotation of satellite images via multifeature joint sparse coding with spatial relation constraint," *IEEE Geosci. Remote Sens. Lett.*, vol. 10, no. 4, pp. 652–656, Jul. 2013.

[34] Z. He, Q. Wang, Y. Shen, and M. J. Sun, "Kernel sparse multitask learning for hyperspectral image classification with empirical mode decomposition and morphological wavelet-based features," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 8, pp. 5150–5163, Aug. 2014.

[35] J. Li, H. Zhang, and L. Zhang, "Efficient superpixel-level multi-task joint sparse representation for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 10, pp. 5338–5351, Oct. 2015.

[36] J. Li, H. Zhang, L. Zhang, X. Huang, and L. Zhang, "Joint collaborative representation with multitask learning for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 9, pp. 5923–5936, Sep. 2014.

[37] J. Li, H. Zhang, and L. Zhang, "A nonlinear multiple features learning classifier for hyperspectral image with limited training samples," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 8, no. 6, pp. 2728–2738, Jun. 2015.

[38] E. Zhang, X. Zhang, H. Liu, and L. Jiao, "Fast multifeature joint sparse representation for hyperspectral image classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 7, pp. 1397–1401, Jul. 2015.

[39] L. Fang, S. Li, X. Kang, and J. A. Benediktsson, "Spectral-spatial hyperspectral image classification via multiscale adaptive sparse representation," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 12, pp. 7738–7749, Dec. 2014.

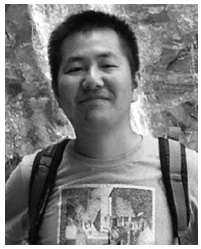
[40] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and M. Yi, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 210–227, Feb. 2009.

[41] H. Zhang, N. M. Nasrabadi, Y. Zhang, and T. S. Huang, "Joint dynamic sparse representation for multi-view face recognition," *Pattern Recognit.*, vol. 45, no. 4, pp. 1290–1298, 2012.

[42] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proc. 5th Annu. Workshop Comput. Learn. Theory*, Pittsburgh, PA, USA, 1992, pp. 144–152.

[43] S. F. Cotter, B. D. Rao, K. Engan, and K. Kreutz-Delgado, "Sparse solutions to linear inverse problems with multiple measurement vectors," *IEEE Trans. Signal Process.*, vol. 53, no. 7, pp. 2477–2488, Jul. 2005.

[44] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4655–4666, Dec. 2007.



Erlei Zhang received the B.S. degree in electrical engineering and automation from Henan Polytechnic University, Jiao Zuo, China, in 2010. He is currently pursuing the Ph.D. degree in pattern recognition and intelligent systems at Xidian University, Xi'an, China.

His research interests include remote sensing image analysis, pattern recognition, and machine learning.

Licheng Jiao (SM'89) received the B.S. degree from Shanghai Jiaotong University, Shanghai, China, in 1982, and the M.S. and Ph.D. degrees from Xi'an Jiaotong University, Xi'an, China, in 1984 and 1990, respectively.

He has been a Professor with the School of Electronic Engineering, Xidian University, Xi'an, since 1992. His current research interests include image processing, natural computation, machine learning, and intelligent information processing.

Xiangrong Zhang (M'07–SM'14) received the B.S. and M.S. degrees in computer science and the Ph.D. degree in electronic engineering from Xidian University, Xi'an, China, in 1999, 2003, and 2006, respectively.

Currently, she is a Professor with the Key Laboratory of Intelligent Perception and Image Understanding of the Ministry of Education, Xidian University. She has been a Visiting Scientist at the Laboratory of Computer Science and Artificial Intelligence, Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, since February 2015. Her research interests include pattern recognition, machine learning, and image analysis and understanding.

Hongying Liu (M'10) received the B.E. and M.S. degrees in computer science and technology from Xi'an University of Technology, Xi'an, China, in 2002 and 2006, respectively, and the Ph.D. degree in engineering from Waseda University, Tokyo, Japan, in 2012.

Currently, she is with the Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education, Xidian University, Xi'an, China. Her research interests include intelligent signal processing, machine learning, compressive sampling, etc.

Shuang Wang (M'07) received the B.S., M.S. and the Ph.D. degrees in circuits and systems from Xidian University, Xi'an, China, in 2000, 2003, and 2007, respectively.

Currently, she is a Professor with the Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education of China, Xidian University. Her research interests include machine learning, image processing and SAR/POLSAR image processing, etc.