Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Joint sparse model-based discriminative K-SVD for hyperspectral image classification

Ziyu Wang[a,c], Jianxiong Liu[b], Jing-Hao Xue[c,*]

[a] Department of Security and Crime Science, University College London, London WC1E 6BT, UK
[b] Intel Programmable Solutions Group, Buckinghamshire HP12 4XF, UK
[c] Department of Statistical Science, University College London, London WC1E 6BT, UK

ARTICLE INFO

Keywords:
Dictionary learning
Classification
Hyperspectral images (HSI)
Joint sparse model (JSM)
Discriminative K-SVD
SOMP

ABSTRACT

Sparse representation classification (SRC) is being widely investigated on hyperspectral images (HSI). For SRC methods to achieve high classification performance, not only is the development of sparse representation models essential, the designing and learning of quality dictionaries also plays an important role. That is, a redundant dictionary with well-designated atoms is required in order to ensure low reconstruction error, high discriminative power, and stable sparsity. In this paper, we propose a new method to learn such dictionaries for HSI classification. We borrow the concept of joint sparse model (JSM) from SRC to dictionary learning. JSM assumes local smoothness and joint sparsity and was initially proposed for classification of HSI. We leverage JSM to develop an extension of discriminative K-SVD for learning a promising discriminative dictionary for HSI. Through a semi-supervised strategy, the new dictionary learning method, termed JSM-DKSVD, utilises all spectrums over the local neighbourhoods of labelled training pixels for discriminative dictionary learning. It can produce a redundant dictionary with rich spectral and spatial information as well as high discriminative power. The learned dictionary can then be compatibly used in conjunction with the established SRC methods, and can significantly improve their performance for HSI classification.

1. Introduction

Sparse representation classification (SRC), proposed in [1], is being widely investigated on hyperspectral images (HSI). It is based on the assumption that high-dimensional data from the same class lie in a low-dimensional subspace. Therefore a signal can be represented by a linear combination of a small number of redundant bases (so-called dictionary atoms). In [2], Chen et al. apply SRC and propose a joint sparse model (JSM) to HSI classification. JSM assumes that all HSI pixels in a small spatial neighbourhood can be jointly approximated by sparse linear combinations of a few common training samples, which can be solved by the simultaneous orthogonal matching pursuit (SOMP) algorithm [3]. However, in JSM all neighbouring pixels make equal contributions to the sparse recovery of the central pixel. To determine more effective neighbours for JSM, several appealing ideas have been proposed [4–8]. In [4], Zhang et al. introduce a non-local approach [5], which assumes that a candidate has its weight determined by the similarity between its neighbourhood and the central pixel's neighbourhood, termed non local weighting (NLW). Tang et al. propose two manifold-based l_1-norm methods, using locally linear embedding and Laplacian eigenmap to regularise local structures of pixels [6]. In [7,8], Fang et al. and Li et al. propose to adopt superpixel methods [9,10] to integrate the spatial structures for JSM. The superpixel is regarded as a small spatial local region which is adaptive in shape and size.

To achieve high classification performance, not only is the development of sparse representation models essential, the designing and learning of quality dictionaries also plays an important role. A well-designed dictionary would have good representation power over a certain sparsity, as well as to support optimal discrimination of classes [11]. Previous literatures have shown that dictionary learning is beneficial to signal representation as well as to classification [12,11,13,14]. In [12], Aharon et al. propose K-SVD, a generalised K-means method, to minimise the signal reconstruction error. It alternates between sparse coding by orthogonal matching pursuit (OMP) [15] and dictionary updating by singular value decomposition (SVD). For face recognition, Zhang et al. introduce into sparse representation a constraint to model classification error [11]. A K-SVD algorithm is then adopted to minimise the sum of reconstruction error and classification error, named as discriminative K-SVD (D-KSVD). In [13], Jiang et al. propose label consistent K-SVD, which incorporates a label-consistent term into D-KSVD, leading to an explicit correspondence between the

* Corresponding author.
E-mail addresses: ziyu.wang.12@ucl.ac.uk (Z. Wang), mydendrobium@hotmail.com (J. Liu), jinghao.xue@ucl.ac.uk (J.-H. Xue).

http://dx.doi.org/10.1016/j.sigpro.2016.10.022
Received 31 December 2015; Received in revised form 28 August 2016; Accepted 27 October 2016
Available online 05 November 2016

dictionary atoms and labels. It also adopts the K-SVD algorithm to solve the optimisation problem. Mairal et al. propose task-driven dictionary learning (TDDL) [14], which is a general formulation for learning sparse representations tuned for specific tasks. TDDL not only can be designed for classification, but also can be designed for regression and compressive sensing.

There have been a limited number of works on developing the dictionary learning algorithms specifically for HSI classification problems. In [7], Fang et al. propose to use a modified class-labelled OMP algorithm in D-KSVD to learn a dictionary of better discriminative power. In [16], Soltani-Farani et al. partition given pixels into contextual groups, and jointly model pixels inside the same contextual group to be in a common subspace. Both methods endeavour to make a better use of the limited amount of labelled training data. Taking one step further, Wang et.al. utilise spatial context of a test pixel within its local neighbourhood to develop a learning vector quantization (LVQ)-based dictionary learning method [17]. In [18], Sun et al. introduce the use of structure information into dictionary learning. They argue that the requirement of a redundant dictionary in sparse coding can be lessened if simultaneous sparse approximation is employed. Therefore they aim to produce a compact dictionary by using a joint or Laplacian sparsity prior and the TDDL framework [14]. Wang et al. follow the same TDDL framework and introduce a more explicitly formulated semi-supervised problem to the compact dictionary learning [19].

In this context, we believe that, in order to develop a dictionary with high discriminative power for HSI classification but from only a limited number of labelled training samples, it is a promising direction to utilise the structure information as much as possible. Considering the discriminative nature of D-KSVD and its imperfection of exploiting spectral signatures only, we think D-KSVD has substantial room to be explored for improvement. Furthermore, we are highly impressed by the recent progress in HSI classification made by the JSM-based algorithms from its leveraging both spectral and spatial information in the representation of HSI pixels. All these factors inspire us to develop a new dictionary learning approach for HSI classification, by enforcing the JSM assumption, of local smoothness and joint sparsity around the limited number of training sample, into D-KSVD through a semi-supervised fashion. In this paper, we propose a new approach called JSM-DKSVD. It is able to capture and organise the rich spectral and spatial information into the learned dictionary, thus offering higher discriminative power for HSI classification tasks.

Experiment results show that, when used in conjunction with established SRC methods, the JSM-DKSVD-trained dictionary can significantly improve the SRC methods' classification performance, and can also outperform state-of-the-art dictionary learning methods for HSI classification.

The main contribution of this research is that we introduce the structure information around a limited number of training pixels into the dictionary learning for HSI, establish a new discriminative optimisation function to jointly model the enriched information, and develop a JSM-constrained D-KSVD algorithm to solve the optimisation problem and produce a desired discriminative dictionary.

2. Joint sparse models for HSI classification

2.1. Sparse Model (SM)

Suppose a B-dimensional pixel, denoted by $\mathbf{x} \in \mathbb{R}^B$, can be approximated by a linear combination of N_D training pixels:

$$\mathbf{x} \approx \mathbf{D}\boldsymbol{\alpha}, \tag{1}$$

where $\mathbf{D} \in \mathbb{R}^{B\times N_D}$ denotes a dictionary constructed by the N_D training pixels (also termed atoms), and $\boldsymbol{\alpha}$ is the N_D-dimensional vector of coefficients in the linear combination.

In a sparse model, $\mathbf{x}$ can be approximated by only a few (e.g. at most L_C) atoms in $\mathbf{D}$. That is, the coefficient vector $\boldsymbol{\alpha}$ is sparse. The values of $\boldsymbol{\alpha}$ can be estimated by solving the following optimisation problem:

$$\hat{\boldsymbol{\alpha}} = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \left\| \mathbf{x} - \mathbf{D}\boldsymbol{\alpha} \right\|_2^2, \ s.\ t. \ \left\| \boldsymbol{\alpha} \right\|_0 \le L_C, \tag{2}$$

where $\|\boldsymbol{\alpha}\|_0$ denotes a l_0-pseudo-norm (i.e. the number of non-zero elements) of $\boldsymbol{\alpha}$, L_C ($L_C \ll N_D$) is defined as the upper bound of the sparsity level of the model. The problem in (2) is NP-hard, but it can be approximately solved by greedy pursuit algorithms such as orthogonal matching pursuit (OMP) [15] or be relaxed by replacing the l_0-pseudo-norm with the l_1-norm. When the problem is solved by OMP, the dictionary $\mathbf{D}$ is column-wise normalised to have unit l_2-norm.

2.2. Joint Sparse Model (JSM)

In HSI, neighbouring pixels in a small area often consist of similar materials and the classes of these materials are few. Hence, local smoothness and sparsity can be assumed for HSI. In JSM [2], it is assumed that all neighbouring pixels around a central pixel share a common sparse pattern. The modelling, learning and labelling for JSM can be described as follows.

Let $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_{T_C}]$, a $B \times T_C$ matrix, denote a small window consisting of T_C pixels and centring on a test pixel $\mathbf{x}_c$, with each pixel $\mathbf{x}_t$ represented by a B-dimensional vector for B spectral bands. The T_C pixels are approximated by sparse linear combinations of atoms from a given dictionary:

$$\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_{T_C}] \approx \mathbf{D}[\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_{T_C}] = \mathbf{D}\mathbf{A}, \tag{3}$$

where $\mathbf{D} \in \mathbb{R}^{B\times N_D}$ is a dictionary with N_D known and labelled atoms, and $\mathbf{A} \in \mathbb{R}^{N_D\times T_C}$ is the matrix of unknown coefficients $[\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_{T_C}]$. Because of the local smoothness and sparsity, we can assume that there are only L_C ($L_C \ll N_D$) non-zero rows in $\mathbf{A}$. This leads to the so-called joint sparse model (JSM), where the non-zero rows form the support shared by coefficient vectors $\{\boldsymbol{\alpha}_t\}_{t=1}^{T_C}$. That is, $\{\boldsymbol{\alpha}_t\}_{t=1}^{T_C}$ are sparse vectors and $\mathbf{A}$ is a sparse matrix.

The learning of JSM is to estimate $\mathbf{A}$, which can be achieved by solving a joint sparse recovery problem:

$$\widehat{\mathbf{A}} = \underset{\mathbf{A}}{\operatorname{argmin}} \|\mathbf{X} - \mathbf{D}\mathbf{A}\|_2^2, \ s.\ t. \ \left\| \mathbf{A} \right\|_{row,0} \le L_C, \tag{4}$$

where $\|\mathbf{A}\|_{row,0}$, the row-wise l_0-norm, is the number of non-zero rows of $\mathbf{A}$. As with (2), problem (4) is NP-hard and it can be approximately solved by greedy algorithms such as the Simultaneous Orthogonal Matching Pursuit algorithm (SOMP) [3] or the Simultaneous Subspace Pursuit algorithm (SSP) [2]. When solved by SOMP or SSP, the dictionary $\mathbf{D}$ is column-wise normalised to have unit l_2-norm.

Once the sparse coefficient matrix $\mathbf{A}$ is obtained, we can calculate the residual (distance) of window $\mathbf{X}$ from its class-wise approximation, which is obtained from using the sub-dictionary of each class, as follows:

$$r^m(\mathbf{X}) = \|\mathbf{X} - \mathbf{D}^m\widehat{\mathbf{A}}^m\|_2^2, \ m = 1, \ldots, M, \tag{5}$$

where M is the total number of classes, $\widehat{\mathbf{A}}^m$ consists of N_m rows in $\widehat{\mathbf{A}}$ that are associated with sub-dictionary $\mathbf{D}^m$ of the mth class, with $N_D = \sum_{m=1}^M N_m$. The label of the central pixel $\mathbf{x}_c$ in window $\mathbf{X}$ is then determined by the minimal residual of $\mathbf{X}$ over all M classes:

$$Class(\mathbf{x}_c) = \underset{m=1,\ldots,M}{\operatorname{argmin}} \ r^m(\mathbf{X}). \tag{6}$$

3. Discriminative dictionary learning algorithms

JSM-based classification methods have achieved improved classification performance over the traditional (individual) sparse model, but most of these methods leave $\mathbf{D}$ simply as a stack of raw labelled pixels

[2,4,6]. On the other hand, the focus of this work is on the learning of **D**. Specifically, we propose to develop a new dictionary learning algorithm, termed JSM-constrained discriminative K-SVD (JSM-DKSVD), to incorporate both the spectral and spatial information into dictionary learning and to improve the performance of HSI classification in the end.

3.1. K-SVD

In K-SVD [12], signals are also represented by their sparse coefficients. It aims to learn a dictionary **D** with unit atoms (bases), which minimises the reconstruction error:

$$\{\widehat{\mathbf{D}}, \widehat{\mathbf{A}}^{train}\} = \underset{\mathbf{D},\mathbf{A}^{train}}{\operatorname{argmin}} \|\mathbf{X}^{train} - \mathbf{D}\mathbf{A}^{train}\|_2^2, \;\; s.\,t. \;\; \|\boldsymbol{\alpha}_p^{train}\|_0 \le L_D, \quad p = 1,\ldots,P, \tag{7}$$

where $\mathbf{D} = [\mathbf{d}_1,\ldots,\mathbf{d}_{N_D}] \in \mathbb{R}^{B\times N_D}$ is a dictionary with N_D atoms to be learned; $\mathbf{X}^{train} = [\mathbf{x}_1^{train},\ldots,\mathbf{x}_P^{train}] \in \mathbb{R}^{B\times P}$ is a training sample set of P training samples; $\mathbf{A}^{train} = [\boldsymbol{\alpha}_1^{train},\ldots,\boldsymbol{\alpha}_P^{train}] \in \mathbb{R}^{N_D\times P}$ is the corresponding sparse coefficient matrix of $\mathbf{X}^{train}$; and L_D ($L_D \ll N_D$) is upper bound of the sparsity level of the model. K-SVD consists of a sparse coding stage and a dictionary updating stage: it first solves (7) with **D** fixed to compute sparse coefficient matrix $\mathbf{A}^{train}$ by the OMP algorithm. Once $\mathbf{A}^{train}$ is obtained, a second stage is performed to update each dictionary atom by SVD one at a time, fixing all other atoms. The two stages are carried out iteratively till certain stopping criteria are met.

3.2. Discriminative KSVD (D-KSVD)

The discriminative K-SVD [11] is proposed to incorporate classification error into the optimisation problem (7), allowing a linear classifier and a dictionary with discriminative power to be learned at the same time.

Specifically, a classification constraint with loss function $\|\mathbf{H}^{train} - \mathbf{W}\mathbf{A}^{train}\|_2^2 + \beta\|\mathbf{W}\|_2^2$ is considered, where $\mathbf{W} = [\mathbf{w}_1,\ldots,\mathbf{w}_{N_D}] \in \mathbb{R}^{M\times N_D}$ is an M-classes linear classifier in the atom space, $\mathbf{H}^{train} = [\mathbf{h}_1^{train},\ldots,\mathbf{h}_P^{train}] \in \mathbb{R}^{M\times P}$ is the class matrix of P training pixels in $\mathbf{X}^{train}$, and $\|\mathbf{W}\|_2^2$ is the regularisation term. Each class vector $\mathbf{h}_p^{train} = [0, 0,\ldots,1,\ldots,0, 0]^T \in \mathbb{R}^M$ corresponds to the labelling of one training sample $\mathbf{x}_p^{train}$ and the non-zero position in $\mathbf{h}_p^{train}$ represents the class of $\mathbf{x}_p^{train}$. The dictionary **D** and the linear classifier **W** are jointly learned by solving the following optimisation problem:

$$\{\widehat{\mathbf{D}}, \widehat{\mathbf{A}}^{train}, \widehat{\mathbf{W}}\} = \underset{\mathbf{D},\mathbf{A}^{train},\mathbf{W}}{\operatorname{argmin}} \{\|\mathbf{X}^{train} - \mathbf{D}\mathbf{A}^{train}\|_2^2 + \gamma\|\mathbf{H}^{train} - \mathbf{W}\mathbf{A}^{train}\|_2^2 + \beta\|\mathbf{W}\|_2^2\}, \;\; s.\,t. \;\; \|\boldsymbol{\alpha}_p^{train}\|_0 \le L_D, \;\; p = 1,\ldots,P, \tag{8}$$

where γ and β control the relative contributions of the corresponding terms. As described in [11], problem (8) can be rewritten as

$$\{\widehat{\mathbf{D}}, \widehat{\mathbf{A}}^{train}, \widehat{\mathbf{W}}\} = \underset{\mathbf{D},\mathbf{A}^{train},\mathbf{W}}{\operatorname{argmin}} \left\{ \left\| \begin{pmatrix} \mathbf{X}^{train} \\ \sqrt{\gamma}\mathbf{H}^{train} \end{pmatrix} - \begin{pmatrix} \mathbf{D} \\ \sqrt{\gamma}\mathbf{W} \end{pmatrix} \mathbf{A}^{train} \right\|_2^2 + \beta\|\mathbf{W}\|_2^2 \right\}, \quad s.\,t. \;\; \|\boldsymbol{\alpha}_p^{train}\|_0 \le L_D, \;\; p = 1,\ldots,P. \tag{9}$$

Following [11], the constraint $\beta\|\mathbf{W}\|_2^2$ is omitted because during the K-SVD process the joint matrix $\begin{pmatrix} \mathbf{D} \\ \sqrt{\gamma}\mathbf{W} \end{pmatrix}$ is always column-wise normalised, i.e. the l_2-norm constraint is implicitly enforced. Now we use the following notation:

$$\mathbf{X}^* = \begin{pmatrix} \mathbf{X}^{train} \\ \sqrt{\gamma}\mathbf{H}^{train} \end{pmatrix}, \; \mathbf{D}^* = \begin{pmatrix} \mathbf{D} \\ \sqrt{\gamma}\mathbf{W} \end{pmatrix}; \tag{10}$$

and problem (9) is approximated by the following optimisation problem:

$$\{\widehat{\mathbf{D}}^*, \widehat{\mathbf{A}}^{train}\} = \underset{\mathbf{D}^*,\mathbf{A}^{train}}{\operatorname{argmin}} \|\mathbf{X}^* - \mathbf{D}^*\mathbf{A}^{train}\|_2^2, \;\; s.\,t. \;\; \|\boldsymbol{\alpha}_p^{train}\|_0 \le L_D, \;\; p = 1,\ldots,P, \tag{11}$$

which can then be solved by the K-SVD algorithm [12].

We note that the obtained matrix $\widehat{\mathbf{D}}^*$ from K-SVD is not the actual dictionary we are looking for. To extract the actual dictionary $\mathbf{D}'$ and the classifier $\mathbf{W}'$, a final normalisation is needed. The dictionary $\mathbf{D}'$ is to be extracted from $\widehat{\mathbf{D}}^*$ and column-wise normalised to have unit l_2-norm; the rest of the matrix $\widehat{\mathbf{D}}^*$, namely classifier $\mathbf{W}'$, is scaled by using the same normalisation constants accordingly:

$$\mathbf{D}' = \left[\frac{\mathbf{d}_1}{\|\mathbf{d}_1\|_2}, \frac{\mathbf{d}_2}{\|\mathbf{d}_2\|_2}, \ldots, \frac{\mathbf{d}_{N_D}}{\|\mathbf{d}_{N_D}\|_2}\right], \quad \mathbf{W}' = \left[\frac{\mathbf{w}_1}{\|\mathbf{d}_1\|_2}, \frac{\mathbf{w}_2}{\|\mathbf{d}_2\|_2}, \ldots, \frac{\mathbf{w}_{N_D}}{\|\mathbf{d}_{N_D}\|_2}\right], \tag{12}$$

where $\mathbf{d}_k$ and $\mathbf{w}_k$ denote the k-th column of **D** and **W**, respectively.

3.3. Classification approach

Given the dictionary $\mathbf{D}'$ and the linear classifier $\mathbf{W}'$, the sparse coefficient vector $\boldsymbol{\alpha}^{test}$ of a test HSI pixel $\mathbf{x}^{test}$ is computed by solving the following problem:

$$\widehat{\boldsymbol{\alpha}}^{test} = \operatorname{argmin}_{\boldsymbol{\alpha}^{test}} \|\mathbf{x}^{test} - \mathbf{D}'\boldsymbol{\alpha}^{test}\|_2^2, \;\; s.\,t. \;\; \|\boldsymbol{\alpha}^{test}\|_0 \le L_C. \tag{13}$$

By applying the linear classifier $\mathbf{W}'$ to $\widehat{\boldsymbol{\alpha}}^{test}$, the class label vector $\mathbf{h}^{test} = [h_1^{test},\ldots,h_M^{test}]^T$ of $\mathbf{x}^{test}$ is obtained as

$$\widehat{\mathbf{h}}^{test} = \mathbf{W}'\widehat{\boldsymbol{\alpha}}^{test}, \tag{14}$$

and the class label of $\mathbf{x}^{test}$ is determined by the position of the maximum value within $\widehat{\mathbf{h}}^{test}$:

$$class(\mathbf{x}^{test}) = \underset{m=1,\ldots,M}{\operatorname{argmax}} \, \widehat{h}_m^{test}. \tag{15}$$

4. JSM-DKSVD

Dictionary learning by K-SVD and D-KSVD only considers spectral signatures of the HSI pixels. Recent developments in JSM-related algorithms show promising results of using not only spectral but also spatial structure information in the representation of pixels. Inspired by this progress, we propose to incorporate the HSI structure information into the dictionary learning process and extend D-KSVD to HSI classification. Specifically, we enforce the assumption of local smoothness of images as well as sparsity of the representations of training HSI pixels into dictionary learning. We name this new dictionary learning approach as JSM-DKSVD.

4.1. Motivation of JSM-DKSVD

The core idea of JSM-DKSVD is to embed the structure information into the representation of dictionary training pixels by joint modelling. The sparse coefficients of a pixel are determined jointly with those in its local neighbourhood, which is a collection of pixels located in a small window centred on the pixel in question. Therefore the training set $\mathbf{X}^{train} = [\mathbf{x}_1^{train},\ldots,\mathbf{x}_P^{train}] \in \mathbb{R}^{B\times P}$ is extended as follows:

$$\mathbf{X}^{JSM} = [X_1^{JSM},\ldots,X_P^{JSM}] \in \mathbb{R}^{B\times(T_D\times P)}, \tag{16}$$

where $X_p^{JSM} \in \mathbb{R}^{B\times T_D}$, $p = 1,\ldots,P$, denotes a small window consisting of T_D pixels and centred on the training pixel $\mathbf{x}_p^{train}$. Each of these neighbourhoods ${X^{JSM}}_p$ is now as a whole to be jointly modelled, replacing the pixel $\mathbf{x}_p^{train}$ in dictionary learning. Note that, although the training sample size is effectively increased from P to $T_D \times P$, as JSM-DKSVD is working in a semi-supervised fashion, we only need P labelled training pixels, which are those central pixels; that is, our JSM-DKSVD method does not require more labelled training samples than

K-SVD and D-KSVD.

The spectral and spatial structure information of a training pixel is therefore exploited by enforcing local smoothness of natural signals, i.e. nearby pixels share a common pattern. In our case, a certain degree of similarity is enforced on the sparse representation patterns of the neighbouring pixels. This forms a new constraint, and will be reflected by expanding the class matrix $\mathbf{H}^{train}$ in D-KSVD correspondingly to a larger matrix $\mathbf{H}^{JSM}$.

If, for example, the central pixel $\mathbf{x}_p^{train}$ in the neighbourhood X_p^{JSM} is labelled class #4 out of five classes, its class vector $\mathbf{h}_p$ in D-KSVD is

$$\mathbf{h}_p = [0, 0, 0, 1, 0]^T, \tag{17}$$

where the non-zero position is at the 4th element. In our JSM-DKSVD, by assuming that the neighbouring pixels share the same class label, the class matrix of the 3×3 window centred on $\mathbf{x}_p$ is as follows:

$$H_p^{JSM} = \begin{bmatrix} 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 1&1&1&1&1&1&1&1&1\\ 0&0&0&0&0&0&0&0&0 \end{bmatrix}_{5\times9}, \tag{18}$$

where the the class vector of each pixel in the window shares the same non-zeros row, i.e. the 4th row of the class matrix $\mathrm{H}^{JSM}{}_p$. Naturally, by concatenating the class matrices of all training pixels $\mathbf{X}^{JSM}$, the overall class matrix $\mathbf{H}^{JSM}$ is

$$\mathbf{H}^{JSM} = [H_1^{JSM}, \ldots, H_P^{JSM}] \in \mathbb{R}^{M\times(T_D\times P)}. \tag{19}$$

4.2. Formulation of JSM-DKSVD

In the proposed JSM-DKSVD, signals in a small neighbourhood are jointly represented by a common sparsity pattern, as in JSM. Meanwhile, a classification constraint with a new class matrix $\mathbf{H}^{JSM}$ is reconstructed, leading to the following optimisation problem:

$$\begin{aligned}\{\widehat{\mathbf{D}}^{JSM}, \widehat{\mathbf{A}}^{JSM}, \widehat{\mathbf{W}}\} = \underset{\mathbf{D}^{JSM},\mathbf{A}^{JSM},\mathbf{W}}{\operatorname{argmin}} \{&\|\mathbf{X}^{JSM} - \mathbf{D}^{JSM}\mathbf{A}^{JSM}\|_2^2 \\ &+ \gamma\|\mathbf{H}^{JSM} - \mathbf{W}\mathbf{A}^{JSM}\|_2^2 + \beta\|\mathbf{W}\|_2^2\},\\ s.\ t.\ \ \|A_p^{JSM}\|_{row,0} \le L_D,\ p = 1,\ldots,P,&\end{aligned} \tag{20}$$

where $\mathbf{X}^{JSM}$ and $\mathbf{H}^{JSM}$ are defined in (16) and (19), respectively; $A_p^{JSM} = [\boldsymbol{\alpha}_{p,1}^{JSM}, \ldots, \boldsymbol{\alpha}_{p,T_D}^{JSM}] \in \mathbb{R}^{N_D\times T_D}$ is the corresponding joint sparse coefficient matrix of a small window $\mathrm{X}_p{}^{JSM}$ as defined in (16), and therefore $\mathbf{A}^{JSM} = [A_1^{JSM}, \ldots, A_P^{JSM}] \in \mathbb{R}^{N_D\times(T_D\times P)}$ is the corresponding sparse coefficient matrix of $\mathbf{X}^{JSM}$; $\mathbf{D}^{JSM} \in \mathbb{R}^{B\times N_D}$ and $\mathbf{W} \in \mathbb{R}^{M\times N_D}$ are the dictionary and classifier to be learned by JSM-DKSVD. This problem can be solved by the K-SVD algorithm. Again, due to the column-wise normalisation through the K-SVD process, the constraint $\beta\|\mathbf{W}\|_2^2$ can be omitted to simplify the problem.

Similarly to (10) and (11), the optimisation problem (20) can be rewritten as

$$\begin{aligned}\{\widehat{\mathbf{D}}^{JSM*}, \widehat{\mathbf{A}}^{JSM}\} &= \underset{\mathbf{D}^{JSM*},\mathbf{A}^{JSM}}{\operatorname{argmin}} \|\mathbf{X}^{JSM*} - \mathbf{D}^{JSM*}\mathbf{A}^{JSM}\|_2^2,\\ s.\ t.\ \ \|A_p^{JSM}\|_{row,0} &\le L_D,\ p = 1,\ldots,P,\end{aligned} \tag{21}$$

where

$$\mathbf{X}^{JSM*} = \begin{pmatrix}\mathbf{X}^{JSM}\\ \sqrt{\gamma}\mathbf{H}^{JSM}\end{pmatrix},\ \text{and}\ \mathbf{D}^{JSM*} = \begin{pmatrix}\mathbf{D}^{JSM}\\ \sqrt{\gamma}\mathbf{W}\end{pmatrix}. \tag{22}$$

Therefore, $\mathbf{X}^{JSM*}$ is a $(B+M)\times(T_D\times P)$ matrix and $\mathbf{D}^{JSM*}$ is a $(B+M)\times N_D$ matrix.

4.3. Algorithm of JSM-DKSVD

The objective function (20) of JSM-DKSVD can be solved by adopting the original K-SVD algorithm in [12], more specifically, by adopting the iterative updating process for $\mathbf{D}^{JSM*}$ and $\mathbf{A}^{JSM*}$.

4.3.1. Initialisation

It is required that the dictionary $\mathbf{D}^{JSM}$ and the classifier $\mathbf{W}$ are given initial values to enable the iterative updating process to follow. Their initial values can be as simple as randomised matrices; in this work we follow the initialisation process of D-KSVD [11] which is explained as follows.

The initial dictionary matrix is denoted by $\mathbf{D}^{(0)}$. As with in [11], $\mathbf{D}^{(0)}$ should have l_2-normalised columns. Given the number of atoms N_D, $\mathbf{D}^{(0)}$ is designed to be a $B\times N_D$ matrix, and it can be initialised by the original K-SVD algorithm with only a couple of iterations.

As a result, the coefficient matrix of $\mathbf{X}^{JSM}$ for initialisation, denoted by $\mathbf{A}^{(0)}$, is computed by solving the first objective term of (20):

$$\mathbf{A}^{(0)} = (\mathbf{D}^{(0)T}\mathbf{D}^{(0)})^{-1}\mathbf{D}^{(0)T}\mathbf{X}^{JSM}, \tag{23}$$

where $\mathbf{A}^{(0)}$ is an $N_D\times(T_D\times P)$ matrix.

The initial classifier, denoted by $\mathbf{W}^{(0)}$, is computed by solving the problem of $\underset{\mathbf{W}^{(0)}}{\operatorname{argmin}}\{\|\mathbf{H}^{JSM} - \mathbf{W}^{(0)}\mathbf{A}^{(0)}\|_2^2 + \|\mathbf{W}^{(0)}\|_2^2\}$:

$$\mathbf{W}^{(0)} = ((\mathbf{A}^{(0)}\mathbf{A}^{(0)T} + \mathbf{I})^{-1}\mathbf{A}^{(0)}\mathbf{H}^{JSM T})^T, \tag{24}$$

where $\mathbf{W}^{(0)}$ is an $M\times N_D$ matrix.

After initialisation, we compose the objective function (21) with $\mathbf{X}^{JSM*} \in \mathbb{R}^{(B+M)\times(T_D\times P)}$ and $\mathbf{D}^{JSM*} \in \mathbb{R}^{(B+M)\times N_D}$, and the iterative updating process of $\mathbf{D}^{JSM*}$ and $\mathbf{A}^{JSM*}$ can start.

4.3.2. Iterative updating - sparse coding stage

Fixing the dictionary $\mathbf{D}^{JSM*}$, we compute the joint sparse coefficient matrix $A_p^{JSM} \in \mathbb{R}^{N_D\times T_D}$ for each training window $X_p^{JSM*} \in \mathbb{R}^{(B+M)\times T_D}$, where $p = 1,\ldots,P$, by approximating the following solution:

$$\widehat{A}_p^{JSM} = \underset{A_p^{JSM}}{\operatorname{argmin}} \|X_p^{JSM*} - \mathbf{D}^{JSM*}A_p^{JSM}\|_2^2,\ \ s.\ t.\ \|A_p^{JSM}\|_{row,0} \le L_D, \tag{25}$$

which can be solved by the SOMP algorithm [2,3,20]. Then the sparse coefficient matrix $\mathbf{A}^{JSM}$ of all training window $\mathbf{X}^{JSM*}$ (16) is concatenated as

$$\mathbf{A}^{JSM} = [A_1^{JSM}, \ldots, A_P^{JSM}] = [\boldsymbol{\alpha}_1^{JSM}, \ldots, \boldsymbol{\alpha}_{T_D\times P}^{JSM}]. \tag{26}$$

4.3.3. Iterative updating – dictionary updating stage

Following the similar idea of SVD in [12], the dictionary is updated atom by atom.

In the jth iteration, for the kth atom $\mathbf{d}_k^{JSM*} \in \mathbb{R}^{B+M}$ in the dictionary $\mathbf{D}^{JSM*(j-1)}$, where $j = 1,\ldots,J$ and $k = 1,\ldots,N_D$; $\mathbf{D}^{JSM*(j-1)}$ is the dictionary obtained from the previous iteration $j-1$, the atom $\mathbf{d}_k^{JSM*}$ is updated to a new one, denoted by $\tilde{\mathbf{d}}_k^{JSM*}$, by the following steps:

a. define a group of the instances that use atom $\mathbf{d}_k^{JSM*}$,

$$\omega_k = \{p|1 \le p \le T_D\times P,\ \mathbf{A}^{JSM}(k, p) \ne 0\}, \tag{27}$$

where $\mathbf{A}^{JSM}(k, p)$ denotes the kth row and pth column of $\mathbf{A}^{JSM}$;

b. compute the overall representation error $\mathbf{E}_k$ without using the atom $\mathbf{d}_k^{JSM*}$,

$$\mathbf{E}_k = \mathbf{X}^{JSM*} - \sum_{i\ne k}\mathbf{d}_i^{JSM*}\mathbf{A}^{JSM}(i,\cdot), \tag{28}$$

where $\mathbf{A}^{JSM}(i,\cdot)$ denotes the ith row of $\mathbf{A}^{JSM}$ and $i = 1,\ldots,N_D$;

c. restrict $\mathbf{E_k}$ by choosing only the column corresponding to ω_k, and obtain $\mathbf{E}_k^R$:

$$\mathbf{E}_k^R = \mathbf{E_k}(\cdot,\omega_k) \tag{29}$$

where $\mathbf{E_k}(\cdot,\omega_k)$ denotes the columns of $\mathbf{E_k}$ corresponding to ω_k;

d. apply SVD decomposition $\mathbf{E}_k^R = \mathbf{U\Delta V^T}$. The updated atom $\tilde{\mathbf{d}}_k^{JSM*}$ and its corresponding sparse coefficients in the updated coefficient matrix $\widetilde{\mathbf{A}}^{JSM*}$ are solved by

$$\widetilde{\mathbf{d}}_k^{JSM*} = \mathbf{U}(\cdot,1)\widetilde{\mathbf{A}}^{JSM*}(k,\omega_k) = \Delta(1,1)\mathbf{V}(\cdot,1) \tag{30}$$

where $\mathbf{U}(\cdot,1)$ and $\mathbf{V}(\cdot,1)$ denotes the first column of $\mathbf{U}$ and the first column of $\mathbf{V}$, respectively.

After J iterations, the desired dictionary $\mathbf{D}'$ and the classifier $\mathbf{W}'$ learned by JSM-DKSVD should also be re-normalised as in (12).

Details of the proposed JSM-DKSVD are summarised in Algorithm 1.

Algorithm 1. JSM-DKSVD algorithm to solve (21).

```
Input:
  Training HSI pixels $\mathbf{X}^{train} \in \mathbb{R}^{B\times P}$.
  Window size $T_D$ for training.
  Control parameter $\gamma$.
  Sparsity level $L_D$.
  Number of atoms $N_D$ of the dictionary to be learned.
  Maximum iteration number $J$.
Output: $\mathbf{D}'$, $\mathbf{W}'$.
Initialisation:
  • Generate training sample set $\mathbf{X}^{JSM}$ by (16).
  • Compose class matrix $\mathbf{H}^{JSM}$ by (19).
  • Initialise dictionary matrix $\mathbf{D}^{(0)}$ with $l_2$-normalised columns.
  • Compute coefficient matrix $\mathbf{A}^{(0)}$ by (23).
  • Initialise classifier $\mathbf{W}^{(0)}$ by (24).
  • Compose $\mathbf{X}^{JSM*}$ and $\mathbf{D}^{JSM*}$ by (22).
  while $j \le J$ do
    Sparse coding stage:
    Compute the sparse coefficient matrix $A_p^{JSM}$ for each training windows $X_p^{JSM*}$ by (25).
    Concatenate the sparse coefficient matrix $\mathbf{A}^{JSM}$ for all training windows $\mathbf{X}^{JSM*}$ by (26).
    Dictionary updating stage:
    for $k$=1:$N_D$ in $\mathbf{D}^{JSM*(j-1)}$ do
      Define the group of instances that use atom $\mathbf{d}_k^{JSM*}$ by (27).
      Compute the overall representation error $\mathbf{E}_k$ by (28).
      Restrict $\mathbf{E}_k$ to $\mathbf{E}_k^R$ by (29).
      Apply SVD decomposition to $\mathbf{E}_k^R$ and solve the updated atom $\widetilde{\mathbf{d}}_k^{JSM*}$ and its corresponding sparse coefficients $\widetilde{\mathbf{A}}^{JSM*}(k,\omega_k)$ by (30).
    end for
    Set $j = j + 1$.
  end while
  Compute the desired dictionary $\mathbf{D}'$ and classifier $\mathbf{W}'$ by (12).
```

4.4. Classification approach of JSM-DKSVD

Same as the D-KSVD, the dictionary $\mathbf{D}'$ and the classifier $\mathbf{W}'$ learned by JSM-DKSVD can be used together with many established HSI classification methods. By embedding richer structure information from HSI in the dictionary and the classifier, the proposed JSM-DKSVD aims to improve the overall classification accuracy when used with these dictionary-based classification methods for HSI.

Specifically, when JSM-DKSVD is used in pair with the JSM-based SRC method, given a test matrix $\mathbf{X}^{test} = [\mathbf{x}_1^{test},\ldots,\mathbf{x}_{T_C}^{test}]$ with T_C pixels centred on the test pixel $\mathbf{x}^{test}$, the JSM coefficient matrix $\mathbf{A}^{test} = [\boldsymbol{\alpha}_1^{test},\ldots,\boldsymbol{\alpha}_{T_C}^{test}]$ is computed by solving the following problem:

$$\widehat{\mathbf{A}}^{test} = \underset{\mathbf{A}^{test}}{\operatorname{argmin}}\|\mathbf{X}^{test} - \mathbf{D}'\mathbf{A}^{test}\|_2^2,\ \ s.\ t.\ \ \|\mathbf{A}^{test}\|_{row,0} \le L_C. \tag{31}$$

Then, the classifier $\mathbf{W}'$ is applied to $\widehat{\mathbf{A}}^{test}$ to create the estimated class label matrix $\widehat{\mathbf{H}}^{test}$ for $\mathbf{X}^{test}$:

$$\widehat{\mathbf{H}}^{test} = \mathbf{W}'\widehat{\mathbf{A}}^{test}. \tag{32}$$

Finally, each row of $\widehat{\mathbf{H}}^{test} \in \mathbb{R}^{M\times T_C}$ is summed together as a new class label vector $\widehat{\mathbf{h}}^{test} \in \mathbb{R}^M$, and the class label of the central test pixel is determined by (15).

5. Experimental studies

The experiments are carried out on two real hyperspectral datasets: the AVIRIS Indian Pines dataset and the ROSIS University of Paiva dataset, both of which are publicly available [21]. We evaluate the proposed JSM-DKSVD and compare the learned dictionary with two other types of dictionaries. The first comparison is against the dictionary constructed by original labelled training pixels, denoted by $\mathbf{D}^{raw}$, such as in [2,4,6]. The second comparison is against the direct application of D-KSVD [11]. Dictionaries acquired from $\mathbf{D}^{raw}$, D-KSVD and the proposed JSM-DKSVD are used with three different SRC methods: 1) SM (referred to as OMP), 2) JSM [2] (referred to as SOMP), and 3) NLW [4].

We employ three standard performance measures for HSI classification: the overall accuracy (OA), the average accuracy (AA) and kappa coefficient κ [22]. The overall accuracy is defined as the ratio of correctly-classified test pixels over all classes; the average accuracy is defined as the average value of the M accuracies of individual classes, where M is the total number of classes; and the kappa coefficient κ is defined as the percentage of classified test pixels corrected by the number of agreements that would be expected purely by chance. The OA, AA and κ are defined as follows:

$$OA = \frac{N_{corr}}{N_{test}},\ \ AA = \frac{1}{M}\sum_{i=1}^{M}\frac{N_i^{corr}}{N_i^{class}},\ \ \kappa = \frac{OA - p_e}{1 - p_e}, \tag{33}$$

where N_{corr} is the number of the correctly-classified test pixels, N_{test} is the total number of test pixels; $N^{corr}{}_i$ is the number of the correctly-classified test pixels of class i, $N^{class}{}_i$ is the total number of test pixels of class i; and $p_e = \sum_{i=1}^{M}(P_i \times P_i^t)$, in which P_i is the ratio of data assigned to class i by the classifier and $P_i{}^t$ is the ratio of data that belong to class i.

The SPAMS toolbox [20] is used to execute the sparse recovery process, i.e. OMP and SOMP. MATLAB codes from [11] are used to perform the K-SVD and D-KSVD algorithms, and MATLAB codes from [4] are used to perform the NLW algorithm.

5.1. Parameter settings

The parameters involved in the whole evaluation process include those for both the SRC methods and the dictionary learning methods. For the SRC methods, the parameters include the sparsity level L_C and the window size T_C for SOMP and NLW. For the dictionary learning methods, the parameters include the sparsity level L_D, the number of atoms N_D, the regularisation parameter γ, the iteration number J, and finally the window size T_D for the proposed JSM-DKSVD method. It is too costly to cross-validate through the entire design space. To simplify the problem, we break it down into two steps:

- When the three SRC methods (OMP, SOMP and NLW) use $\mathbf{D}^{raw}$, we perform the leave-one-out-cross-validation (LOOCV) to tune their parameters L_C and T_C. Then the parameters of the three SRC methods are fixed and decoupled from dictionary learning, providing a relatively fair testing platform for the dictionary learning methods.
- For the D-KSVD and JSM-DKSVD dictionary learning methods which produce dictionaries that are independent of and universally applicable to different SRC methods, we define their parameters in the following way. Parameter N_D by the nature cannot be tuned by

Table 1
The Indian Pines dataset: Ground-truth labels, class material, the training set and the test set. The middle two columns are for the case of 957 training pixels (9% of all pixels) and 9409 test pixels; the rightmost two columns are for the case of 524 training pixels (5% of all pixels) and 9842 test pixels.

Class	Material	Training	Test	Training	Test
1	Alfalfa	5	49	3	51
2	Corn-notill	132	1302	72	1362
3	Corn-mintill	77	757	42	792
4	Corn	22	212	12	222
5	Grass-pasture	46	451	25	472
6	Grass-trees	69	678	38	709
7	Grass-pasture-mowed	3	23	2	24
8	Hay-windrowed	45	444	25	464
9	Oats	2	18	1	19
10	Soybean-notill	89	879	49	919
11	Soybean-mintill	227	2241	124	2344
12	Soybean-clean	57	557	31	583
13	Wheat	20	192	11	201
14	Woods	119	1175	65	122
15	Buildings-grass-trees-drives	35	345	19	361
16	Stone-steel-towers	9	86	5	90
Total		957	9409	524	9842

cross-validation. Therefore, we set N_D to be the maximum possible number of atoms, which is dataset-dependent (see details in the following sections). Empirically and for simplicity, the regularisation parameter γ is set to be 1 and the iteration number J is set to be 30. The sparsity level L_D for the matching pursuit algorithms is set to be 5 and 30 respectively for DKSVD and JSM-DKSVD due to the difference between OMP and SOMP. Finally, we evaluate JSM-DKSVD with two training window sizes, 3×3 and 5×5, for illustrative purposes.

5.2. AVIRIS dataset: Indian Pines

The AVIRIS Indian Pines dataset consists of 145×145 pixels from 224 spectral bands with sixteen ground-truth labels. Similarly to [2,4], we use its 200 bands after removing the water absorption bands. Following [4], we first randomly choose 957 labelled pixels (9.23% of all pixels) for training, i.e. $\mathbf{X}^{train} \in \mathbb{R}^{200\times957}$. The rest pixels are used for testing, i.e. $\mathbf{X}^{test} \in \mathbb{R}^{200\times9409}$. A summary of the numbers of training and test pixels for individual classes is given in the middle two columns in Table 1. The sixteen ground-truth classes, the training set as well as the test set are shown in Figs. 1(a)–(c).

For the three SRC methods, OMP, SOMP and NLW using $\mathbf{D}^{raw}$, the optimal parameters obtained by LOOCV are L_C=5 for OMP, L_C=30 and $T_C = 7\times7$ for SOMP and L_C=30 and $T_C = 9\times9$ for NLW.

Regarding the number of atoms, we set N_D=957 for D-KSVD. For the JSM-DKSVD dictionary learning method, due to the possible overlapping of the extended neighbourhoods, its training set $\mathbf{X}^{JSM}$, which is the extended $\mathbf{X}^{train}$, may not be full rank and as a result the K-SVD algorithm cannot be executed. The maximum possible number of atoms for JSM-DKSVD is therefore defined to be the maximum unique columns of $\mathbf{X}^{JSM}$. For the training window $T_D = 3\times3$, the unique number of atoms is 5,145; and for $T_D = 5\times5$, the unique number of atoms is 8,764. Therefore we set $N_D = 5,145$ and $N_D = 8,764$ under $T_D = 3\times3$ and $T_D = 5\times5$, respectively.

To have a reliable evaluation and fair comparison, we repeat the experiments for 20 times under these parameter settings through performing 20 random training-test splits. For each of the 12 combinations of four dictionary learning schemes and three SRC methods with their optimal parameters, all of its 20 overall classification accuracies are recorded and box-plotted in Fig. 2(a). Moreover, for illustrative purposes, the classification results for one experiment randomly selected from the 20 experiments are given in Table 2 and depicted in Figs. 3(a)–(l), respectively.

It can be observed that, firstly, the D-KSVD method does not improve the classification performance significantly, compared with those simply using $\mathbf{D}^{raw}$ for HSI classification. Secondly, in contrast to D-KSVD, JSM-DKSVD is capable of producing a dictionary-classifier combination of much better performance than the other two dictionary learning methods in both cases of $T_D = 3\times3$ and $T_D = 5\times5$. In Table 2, for OMP, the overall accuracy (OA) is improved the most, with an 11% (78.63–89.94%) increase under $T_D = 3\times3$ and with a 14% (78.63–92.99%) increase under $T_D = 5\times5$. For SOMP and NLW, OAs are also improved, by around 4% (93.85–97.95% and 95.00–98.68%) under $T_D = 3\times3$. JSM-DKSVD combined with NLW reaches the highest accuracies, 98.68%.

To further demonstrate the benefit of using the JSM-DKSVD-trained dictionary, the same test is performed again but with even fewer training pixels. For this test, only around 5% of the total pixels are chosen as training pixels, i.e. $\mathbf{X}^{train} \in \mathbb{R}^{200\times524}$, and the rest of the pixels are used for testing, i.e. $\mathbf{X}^{test} \in \mathbb{R}^{200\times9842}$. The summarised dataset is shown in the rightmost two columns in Table 1. The number of atoms N_D is set by the same process as that in the case of 9% training pixels, and the results are 2,919 under $T_D = 3\times3$ and 5,831 under $T_D = 5\times5$, respectively. Parameters L_C, T_C, L_D, T_D, γ and J are remained the same as those in 9% pixels for training.

We randomly split the dataset into training-test pairs for 20 times. All of the 20 overall classification accuracies are shown in Fig. 2. The classification results for one experiment randomly selected from the 20 experiments are shown in Table 3, excluding the classification accuracies of individual classes and the classification maps to save space. Once again, JSM-DKSVD-trained dictionaries are still capable of improving the performance of the reference SRC methods to a high standard, and can be much superior to the SRC methods with DKSVD-trained dictionaries.

We also compare our proposed JSM-DKSVD method with state-of-the-art method proposed in [18], which also incorporates the structure

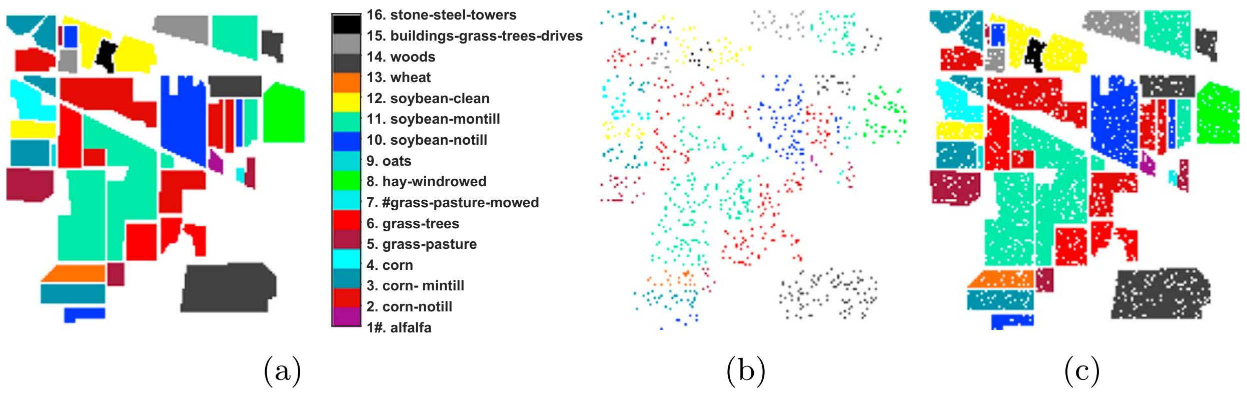

Fig. 1. The Indian Pines dataset with 9% pixels randomly chosen for training: (a) ground-truth labels; (b) training set; (c) test set.

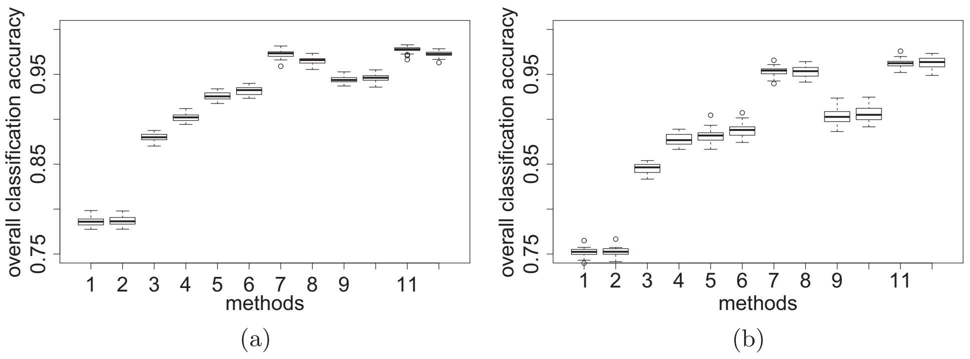

Fig. 2. Boxplots of the overall classification accuracies for the Indian Pines dataset, for 12 combinations indexed by the horizontal axis: (1) $\mathbf{D}^{raw}$-OMP, (2) DKSVD-OMP, (3) JSM-DKSVD-OMP under $T_D = 3 \times 3$, (4) JSM-DKSVD-OMP under $T_D = 5 \times 5$, (5) $\mathbf{D}^{raw}$-SOMP, (6) DKSVD-SOMP, (7) JSM-DKSVD-SOMP under $T_D = 3 \times 3$, (8) JSM-DKSVD-SOMP under $T_D = 5 \times 5$, (9) $\mathbf{D}^{raw}$-NLW, (10) DKSVD-NLW, (11) JSM-DKSVD-NLW under $T_D = 3 \times 3$, and (12) JSM-DKSVD-NLW under $T_D = 5 \times 5$. Each boxplot is constructed from the results of 20 experiments, with panel (a) for the case that 9% pixels are randomly chosen to train the dictionary; and panel (b) for the case that 5% pixels are randomly chosen to train the dictionary.

Table 2
The classification accuracy (%) for the Indian Pines dataset with 957 training pixels (9% of all pixels) and 9409 test pixels, of four dictionary learning methods ($\mathbf{D}^{raw}$, DKSVD, JSM-DKSVD (3×3), and JSM-DKSVD (5×5)) for three SRC methods (OMP, SOMP, NLW). T_D: training window size; N_D: number of atoms; OA: overall accuracy (%); AA: average accuracy (%); κ: kappa coefficient.

	$\mathbf{D}^{raw}$			DKSVD			JSM-DKSVD					
T_D	N/A			N/A			3×3			5×5		
N_D	957			957			5145			8764		
	OMP	SOMP	NLW	OMP	SOMP	NLW	OMP	SOMP	NLW	OMP	SOMP	NLW
1	61.22	79.59	93.88	63.27	77.55	91.84	87.76	91.84	97.96	95.92	100.00	97.96
2	70.35	92.93	94.62	70.28	92.32	94.01	87.17	97.85	98.31	96.08	98.62	98.77
3	65.65	85.73	87.71	65.39	86.79	88.38	86.92	97.62	98.55	91.94	98.02	97.49
4	54.25	89.15	85.38	53.77	89.62	84.43	73.11	99.06	98.58	84.43	97.17	97.64
5	94.24	96.45	98.45	94.46	96.67	98.67	98.23	99.78	99.78	98.89	98.23	100.00
6	94.99	99.26	99.85	94.99	99.85	99.85	97.20	99.56	99.71	97.79	98.53	99.56
7	56.52	56.52	30.43	56.52	56.52	26.09	91.30	82.61	86.96	78.26	73.91	43.48
8	96.62	100.00	100.00	96.40	100.00	100.00	99.32	100.00	100.00	99.77	100.00	100.00
9	55.56	16.67	16.67	55.56	16.67	16.67	77.78	44.44	50.00	61.11	0.00	5.56
10	63.82	77.82	82.48	63.82	79.64	82.82	85.89	93.86	96.36	86.58	94.43	94.65
11	79.43	95.67	98.39	79.70	96.97	98.62	86.93	98.30	99.06	89.60	97.72	98.88
12	72.53	93.00	97.13	72.35	96.23	97.85	86.54	97.49	99.10	91.92	95.69	97.13
13	99.48	98.96	99.48	99.48	100.00	99.48	98.96	99.48	98.96	85.94	84.90	95.31
14	94.47	97.19	97.45	94.47	97.36	97.36	97.62	99.15	99.23	98.30	99.40	100.00
15	57.39	97.68	97.97	57.68	98.55	99.71	85.22	100.00	99.71	93.91	99.71	98.84
16	82.56	98.84	98.84	83.72	98.84	98.84	89.53	93.02	97.67	81.40	83.72	87.21
OA	78.58	93.05	94.89	78.63	93.85	95.00	89.94	97.95	**98.68**	92.99	97.28	98.02
AA	74.94	85.97	86.17	75.12	86.47	85.91	89.34	93.38	**95.00**	89.49	88.75	88.28
κ	0.755	0.921	0.943	0.756	0.923	0.943	0.885	0.977	**0.985**	0.920	0.969	0.978

information into their dictionary learning processes. Referenced directly from [18], the test environment is slightly different in that 997 pixels (10.64% of all 16-classes pixels) are used for training (comparing to the 957 pixels in our case). Under the two similar test settings, our proposed JSM-DKSVD outperforms (98.68% as shown in Table 2) the best performance in [18] which is 94.20%. It is worth noting though: the method proposed in [18] aims to train compact dictionaries and therefore is still expected to have an edge on the computational cost.

The two parameters L_D and N_D in the dictionary learning process are essential to the quality of the resultant dictionary. To better investigate this matter for the proposed JSM-DKSVD, a sweep of the parameter space of N_D and L_D is performed, using 5% pixels for training and the training window $T_D = 3 \times 3$, for example. The classification accuracies of OMP, SOMP and NLW with JSM-DKSVD-trained dictionaries are depicted in Fig. 4.

In all OMP (Fig. 4(a)), SOMP (Fig. 4(b)) and NLW (Fig. 4(c)) settings, it can be seen that the performances of the learned dictionary are consistently maximal when the number of atoms N_D is approaching the maximum value. In these cases, the dictionary is large and flexible enough to store the rich information provided by the extended training neighbourhoods in JSM-DKSVD.

When N_D drops below 1800, the performance becomes unstable, with local maximal observed in different places depending on L_D. This is because: although the dictionaries in these cases are not big enough to support excellent representation of the training neighbourhoods themselves, when the sparsity level L_D is appropriately matched, the resultant dictionary can still achieve a relatively good performance.

Summarising all the L_D dimensions, Fig. 5 shows the best performance that the dictionary can achieve under different N_D. It can be seen that despite of the local maximum mentioned above, the best performance remains at the places where N_D is close to the number of unique columns of $\mathbf{X}^{JSM}$.

Moreover, we can observe that the performance of the learned dictionary is not sensitive to L_D when N_D is approaching the maximum value. Therefore, based on the above discussion we can take the

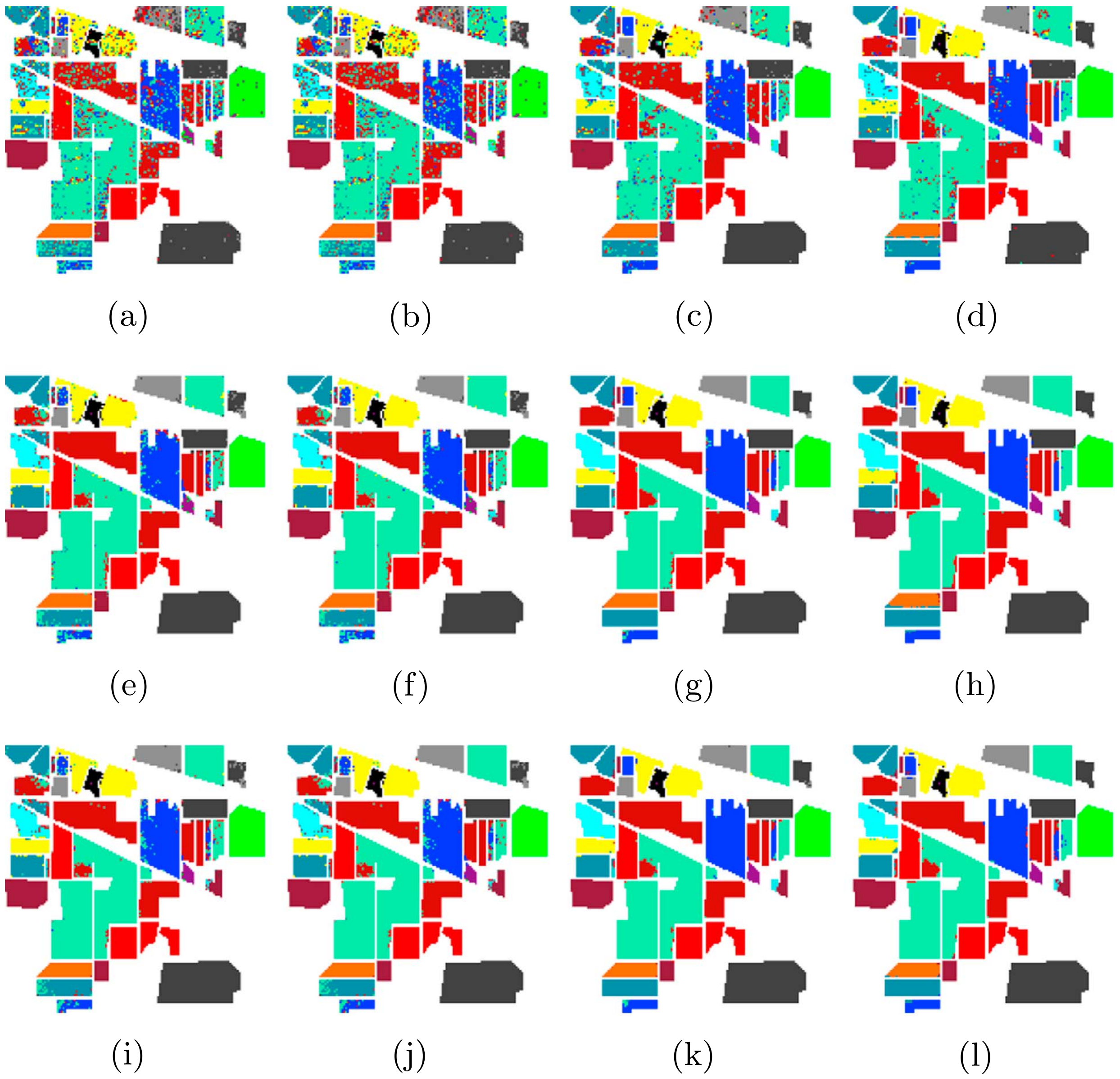

Fig. 3. The classification maps of the Indian Pines dataset with 9% pixels randomly chosen for training: (a) $\mathbf{D}^{raw}$-OMP; (b) DKSVD-OMP (c) JSM-DKSVD-OMP (3×3); (d) JSM-DKSVD-SOMP (5×5); (e) $\mathbf{D}^{raw}$-SOMP; (f) DKSVD-SOMP (g) JSM-DKSVD-SOMP (3×3); (h) JSM-DKSVD-SOMP (5×5); (i) $\mathbf{D}^{raw}$-NLW; (j) DKSVD-NLW (k) JSM-DKSVD-NLW (3×3); (l) JSM-DKSVD-NLW (5×5).

strategy of setting N_D to be close to the number of unique columns in $\mathbf{X}^{JSM}$ and giving L_D certain flexibility.

5.3. ROSIS dataset: University of Pavia

The ROSIS University of Pavia dataset consists of 610×340 pixels from 103 spectral bands, with nine ground-truth labels. We randomly choose only 1% of labelled samples for training, i.e. $\mathbf{X}^{train} \in \mathbb{R}^{103\times432}$ and the rest for testing, i.e. $\mathbf{X}^{test} \in \mathbb{R}^{103\times42344}$. A summary of this dataset is given in Table 4. The nine ground-truth classes, the training set as well as the test set are shown in Figs. 6(a)–(c).

For the three SRC methods, OMP, SOMP, NLW using $\mathbf{D}^{raw}$, the

Table 3
The overall classification accuracy (%) on the Indian Pines dataset with 524 training pixels (5% of all pixels) and 9842 test pixels. The notation is as for Table 2.

	$\mathbf{D}^{raw}$			DKSVD			JSM-DKSVD					
T_D	N/A			N/A			3×3			5×5		
N_D	524			524			2919			5831		
	OMP	SOMP	NLW	OMP	SOMP	NLW	OMP	SOMP	NLW	OMP	SOMP	NLW
OA	75.75	90.46	92.35	75.67	91.15	92.48	85.17	96.08	96.97	88.41	95.74	**97.02**
AA	70.12	82.37	84.04	70.28	82.33	83.46	83.56	93.31	91.52	87.01	93.46	**93.47**
κ	0.723	0.891	0.913	0.722	0.899	0.914	0.831	0.955	0.966	0.868	0.952	**0.966**

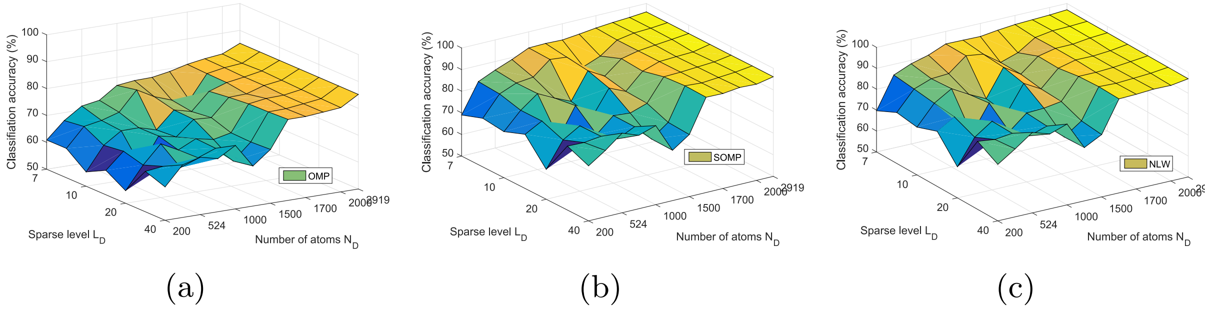

Fig. 4. The overall classification accuracies of using JSM-DKSVD with different numbers of atoms N_D and training sparsity levels L_D. The 5% pixels randomly chosen from the Indian Pines dataset are used to train dictionaries under $T_D = 3 \times 3$. The three SRC methods for testing are (a) OMP, (b) SOMP, and (c) NLW.

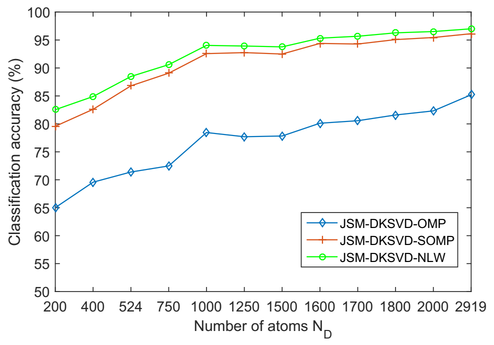

Fig. 5. The optimal classification accuracies of OMP, SOMP, and NLW using the JSM-DKSVD-trained dictionary with different numbers of atoms N_D. The 5% pixels randomly chosen from the Indian Pines dataset are used to train the dictionaries under $T_D = 3 \times 3$.

Table 4
The Pavia University dataset: Ground-truth labels, class material, the training set and the test set.

Class	materials	Training	Test
1	Asphalt	67	6564
2	Meadows	187	18462
3	Gravel	21	2078
4	Trees	31	3033
5	Painted metal sheets	14	1331
6	Bare soil	51	4978
7	Bitumen	14	1316
8	Self-blocking bricks	37	3645
9	Shadows	10	937
Total		432	42344

optimal parameters obtained by LOOCV are L_C=5 for OMP, L_C=10 and $T_C = 3 \times 3$ for SOMP and L_C=20 and $T_C = 5 \times 5$ for NLW.

For the D-KSVD and JSM-DKSVD algorithms, we set the number of atoms N_D=432 for D-KSVD. For the JSM-DKSVD, the unique number of atoms is 3,604 under the training window $T_D = 3 \times 3$, and 9,344 under the training window $T_D = 5 \times 5$. Therefore N_D is set as 3,604 and 9,344 under $T_D = 3 \times 3$ and $T_D = 5 \times 5$, respectively.

As with Section 5.2, we randomly split the dataset into training-test pairs for 20 times. All of the 20 overall classification accuracies are box-plotted in Fig. 7. The classification results for one experiment random selected from the 20 experiments are shown in Table 5 and Figs. 8(a)–(l). Once again, we can observe that the JSM-DKSVD-trained dictionary combined with the three SRC methods outperforms the other two methods ($\mathbf{D}^{raw}$ and D-KSVD) in both cases of $T_D = 3 \times 3$ and $T_D = 5 \times 5$.

Again, we compare our results against those in [18] which is a state-of-the-art dictionary learning method. Even with only 1% of the pixels used for training, the proposed JSM-DKSVD method achieves higher OA (92.77% as shown in Table 5) than that reported (85.70%) in [18], which is evaluated with 10% pixels as training pixels.

5.4. Discussion

JSM-DKSVD utilises the JSM constraint in a fundamentally different way from JSM-based classification methods (SOMP and NLW): JSM-DKSVD applies its constraint to dictionary learning while SOMP and NLW apply their constraints to classification. Moreover, the JSM constraint in classification is used to ensure stable sparse representation for the test pixels when they are classified, while the JSM constraint in JSM-DKSVD is used to ensure richer spectral and spatial information incorporated into the learned dictionary.

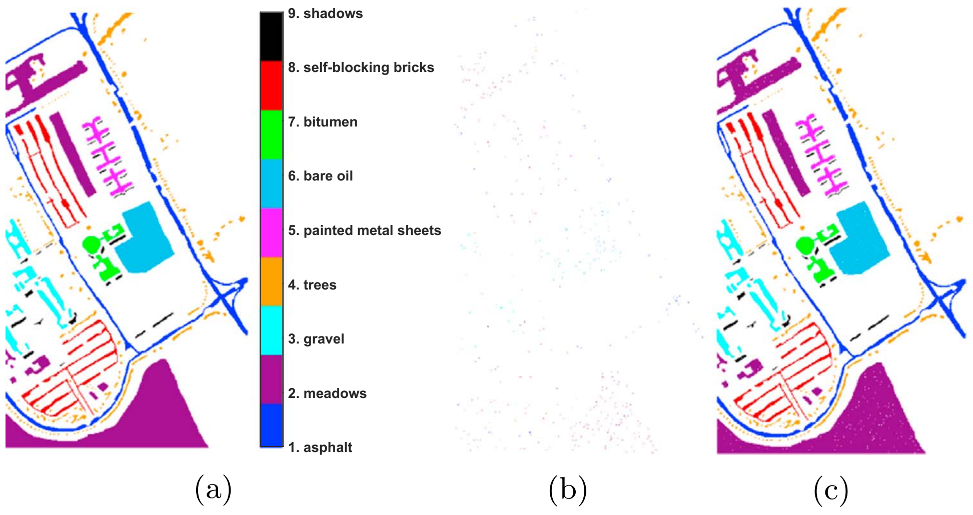

Fig. 6. The University of Pavia dataset with 1% pixels randomly chosen for training: (a) ground-truth labels; (b) training set; (c) test set.

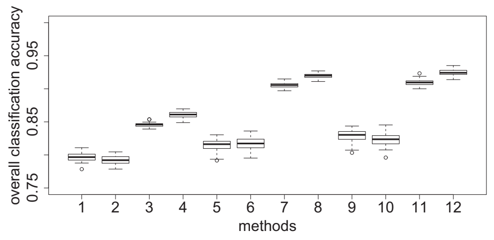

Fig. 7. Boxplots of the overall classification accuracies the University of Pavia dataset: (1) $\mathbf{D}^{raw}$-OMP, (2) DKSVD-OMP, (3) JSM-DKSVD-OMP under $T_D = 3 \times 3$, (4) JSM-DKSVD-OMP under $T_D = 5 \times 5$, (5) $\mathbf{D}^{raw}$-SOMP, (6) DKSVD-SOMP, (7) JSM-DKSVD-SOMP under $T_D = 3 \times 3$, (8) JSM-DKSVD-SOMP under $T_D = 5 \times 5$, (9) $\mathbf{D}^{raw}$-NLW, (10) DKSVD-NLW, (11) JSM-DKSVD-NLW under $T_D = 3 \times 3$, and (12) JSM-DKSVD-NLW under $T_D = 5 \times 5$. Each boxplot is constructed from the results of 20 experiments and 1% pixels are randomly chosen to train the dictionary.

assumption and may prefer a small window for dictionary learning and classification. As we can observe, applying a stronger JSM-constraint during dictionary learning by switching from $T_D = 3 \times 3$ to a larger window $T_D = 5 \times 5$ actually results in a drop of performance of all three classification methods (OMP, SOMP, and NLW) for class 7 and class 9. The optimal choice of the window size (e.g. T_D) can be data-dependent, as is the case for SOMP and NLW where the JSM-constraint is employed for classification and for JSM-DKSVD where the constraint is employed for dictionary learning. It is indeed of our research interests to further investigate and make the window selection process data-adaptive for JSM-DKSVD.

Finally for reference purposes, we discuss the time cost for training dictionaries. All experiments are performed on Xeon E5-1650 CPU (single thread). Table 6 lists the execution time of training dictionaries by D-KSVD, JSM-DKSVD (3×3) and JSM-DKSVD (5×5) conducted at their optimal parameters for the Pavia dataset with 1% pixels randomly chosen for training. The execution time (sec/atom) is normalised by the numbers of trained atoms, i.e. 432 in D-KSVD and 3604 in JSM-DKSVD, respectively. Firstly, it should be noted that there is no training phase on $\mathbf{D}^{raw}$ since the atoms of the dictionary are constructed directly from the training pixels. Secondly, JSM-DKSVD spends more time than D-KSVD for both window sizes, i.e. $T_D = 3 \times 3$ and 5×5, and JSM-DKSVD (5×5) spends the most. These are expected, as the extra cost comes from the neighbours involved with JSM in the training phase. This time/dictionary quality trade-off is often preferred for offline training, which is not uncommon in the literature of the HSI classification.

Table 5
The classification accuracy (%) on the University of Pavia dataset with 432 training pixels (1% of all pixels) and 42344 test pixels. The notation is as for Table 2.

	$\mathbf{D}^{raw}$			D-KSVD			JSM-DKSVD					
T_D	N/A			N/A			3×3			5×5		
N_D	432			432			3604			9344		
	OMP	SOMP	NLW	OMP	SOMP	NLW	OMP	SOMP	NLW	OMP	SOMP	NLW
1	74.21	78.63	83.27	78.47	86.11	90.65	83.68	90.33	93.19	85.33	90.36	93.95
2	92.32	97.82	98.02	91.12	97.36	98.02	91.63	98.00	98.44	92.40	98.34	98.40
3	52.21	63.72	61.50	51.68	61.79	56.79	65.69	76.23	77.33	71.22	79.79	83.16
4	84.54	88.39	88.99	83.42	86.84	84.01	85.86	89.45	89.55	89.28	92.78	92.09
5	99.55	100.00	100.00	95.94	99.02	99.25	98.42	99.85	99.62	98.87	99.92	99.47
6	58.54	63.00	60.81	58.28	64.34	62.56	68.74	74.93	74.89	74.21	81.86	82.74
7	73.25	88.83	89.21	72.64	88.15	86.85	72.64	88.91	87.84	77.20	90.58	91.19
8	65.54	76.19	75.23	59.34	72.18	69.66	67.54	80.80	76.87	71.22	81.59	78.68
9	81.00	81.75	81.96	92.53	97.01	94.77	95.73	98.40	98.51	92.32	97.55	97.65
OA	80.10	85.97	86.39	79.68	86.83	86.86	83.66	90.72	91.04	85.81	92.20	**92.77**
AA	75.69	82.04	82.11	75.94	83.65	82.51	81.10	88.54	88.47	83.56	90.31	**90.82**
κ	0.734	0.811	0.816	0.729	0.823	0.822	0.783	0.876	0.880	0.812	0.896	**0.903**

As dictionary learning can be treated as a pre-processing step for the subsequent classification process and the learned dictionary can be utilised by any sparse representation-based classifiers, JSM-DKSVD can be compatibly utilised in conjunction with existing JSM-based or non-JSM-based classification methods, such as SOMP, NLW and OMP. Because of the difference in the use of the JSM constraint, such a combination of dictionary learning and classification will not introduce undesirable over-smoothness.

Nevertheless, it is worth noting that the JSM constraint itself, be it executed in the dictionary learning or classification process, is based on the grand assumption of signal continuity in natural images. This assumption may be violated in certain part of an image in practice. The violation might be caused by the low resolution of capturing devices, by the very existence of pixels near the border of object regions, or simply by the effect of random noises. This limits the performance of all dictionary learning/classification methods that are based on this assumption.

For example, as is shown in Table 2 for class 7 and class 9 of the Indian Pine dataset, the OMP method, which is not based on the JSM assumption, in fact achieves higher classification accuracies than SOMP and NLW, which are JSM-based. We note that both class 7 and class 9 are small regions with only 26 and 20 pixels in total, respectively (shown in Table 1).

In fact, such a small class is prone to violate the smoothness

6. Conclusion

In this paper, we have proposed a novel dictionary learning method called JSM-DKSVD for hyperspectral image classification. Based on the concept of joint sparse modelling, we incorporate spectral and spatial structure information into the process of discriminative K-SVD, which results in a more informative and discriminative dictionary. Experiment results demonstrate that the proposed JSM-DKSVD achieves better classification performance than those using established dictionary construction methods, even when only a very small fraction (1% for example) of the pixels from the benchmark HSI are used for training.

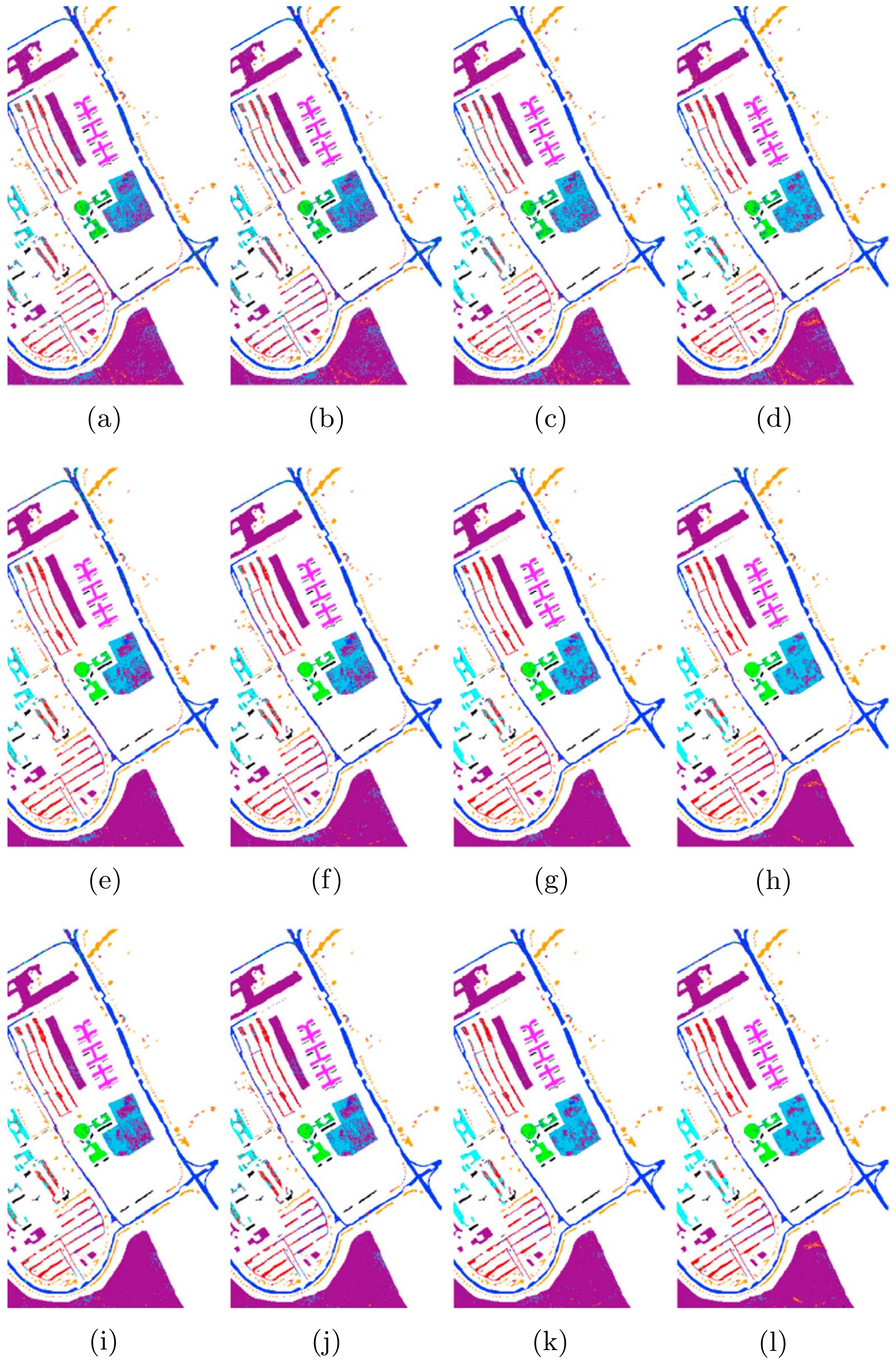

Fig. 8. The classification maps of the University of Pavia dataset with 1% pixels randomly chosen for training: (a) $\mathbf{D}^{raw}$-OMP; (b) DKSVD-OMP (c) JSM-DKSVD-OMP (3×3); (d) JSM-DKSVD-SOMP (5×5); (e) $\mathbf{D}^{raw}$-SOMP; (f) DKSVD-SOMP (g) JSM-DKSVD-SOMP (3×3); (h) JSM-DKSVD-SOMP (5×5); (i) $\mathbf{D}^{raw}$-NLW; (j) DKSVD-NLW (k) JSM-DKSVD-NLW (3×3); (l) JSM-DKSVD-NLW (5×5).

Table 6
Execution time (sec/atom) spent on the University of Pavia dataset with 432 training pixels (1% of all pixels) for training dictionaries.

	$\mathbf{D}^{raw}$	D-KSVD	JSM-DKSVD (3×3)	JSM-DKSVD (5×5)
Time		0.014	0.245	0.512

Acknowledgements

The authors would like to thank Dr. Hongyan Zhang and Dr. Qiang Zhang for providing the software for the non-local weighting (NLW) method and the discriminative K-SVD method, respectively. This work was partially supported by University College London's Security Science Doctoral Training Centre under Engineering and Physical Sciences Research Council (EPSRC) grant EP/G037264/1.

References

[1] J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry, Y. Ma, Robust face recognition via sparse representation, IEEE Trans. Pattern Anal. Mach. Intell. 31 (2) (2009) 210–227.

[2] Y. Chen, N.M. Nasrabadi, T.D. Tran, Hyperspectral image classification using dictionary-based sparse representation, IEEE Trans. Geosci. Remote Sens. 49 (10) (2011) 3973–3985.

[3] J.A. Tropp, A.C. Gilbert, M.J. Strauss, Algorithms for simultaneous sparse approximation. Part I: greedy pursuit, Signal Process. 86 (3) (2006) 572–588.

[4] H. Zhang, J. Li, Y. Huang, L. Zhang, A nonlocal weighted joint sparse representation classification method for hyperspectral imagery, IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 7 (6) (2014) 2056–2065.

[5] J. Mairal, F. Bach, J. Ponce, G. Sapiro, A.Zisserman, Non-local sparse models for image restoration, in: Computer Vision, 2009 IEEE Proceedings of the 12th International Conference on, IEEE, 2009, pp. 2272–2279.

[6] Y.Y. Tang, H. Yuan, L. Li, Manifold-based sparse representation for hyperspectral image classification, IEEE Trans. Geosci. Remote Sens. 52 (12) (2014) 7606–7618.

[7] L. Fang, S. Li, X. Kang, J.A. Benediktsson, Spectral-spatial classification of hyperspectral images with a superpixel-based discriminative sparse model, IEEE Trans. Geosci. Remote Sens. 53 (8) (2015) 4186–4201.

[8] J. Li, H. Zhang, L. Zhang, Efficient superpixel-level multitask joint sparse representation for hyperspectral image classification, IEEE Trans. Geosci. Remote Sens. 53 (10) (2015) 5338–5351.

[9] M.-Y. Liu, O. Tuzel, S. Ramalingam, R. Chellappa, Entropy rate superpixel segmentation, in: Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, IEEE, 2011, pp. 2097–2104.

[10] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, S. Susstrunk, SLIC superpixels compared to state-of-the-art superpixel methods, IEEE Trans. Pattern Anal. Mach. Intell. 34 (11) (2012) 2274–2282.

[11] Q. Zhang, B. Li, Discriminative K-SVD for dictionary learning in face recognition, in: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, IEEE, 2010, pp. 2691–2698.

[12] M. Aharon, M. Elad, A. Bruckstein, K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation, IEEE Trans. Signal Process. 54 (11) (2006) 4311–4322.

[13] Z. Jiang, Z. Lin, L.S. Davis, Label consistent K-SVD: learning a discriminative dictionary for recognition, IEEE Trans. Pattern Anal. Mach. Intell. 35 (11) (2013) 2651–2664.

[14] J. Mairal, F. Bach, J. Ponce, Task-driven dictionary learning, IEEE Trans. Pattern Anal. Mach. Intell. 34 (4) (2012) 791–804.

[15] J. Tropp, A.C. Gilbert, et al., Signal recovery from random measurements via orthogonal matching pursuit, IEEE Trans. Inf. Theory 53 (12) (2007) 4655–4666.

[16] A. Soltani-Farani, H. Rabiee, S. Hosseini, Spatial-aware dictionary learning for hyperspectral image classification, IEEE Trans. Geosci. Remote Sens. 53 (1) (2015) 527–541.

[17] Z. Wang, N.M. Nasrabadi, T.S. Huang, Spatial-spectral classification of hyperspectral images using discriminative dictionary designed by learning vector quantization, IEEE Trans. Geosci. Remote Sens. 52 (8) (2014) 4808–4822.

[18] X. Sun, N.M. Nasrabadi, T.D. Tran, Task-driven dictionary learning for hyperspectral image classification with structured sparsity constraints, IEEE Trans. Geosci. Remote Sens. 53 (8) (2015) 4457–4471.

[19] Z. Wang, N.M. Nasrabadi, T.S. Huang, Semisupervised hyperspectral classification using task-driven dictionary learning with laplacian regularization, IEEE Trans. Geosci. Remote Sens. 53 (3) (2015) 1161–1173.

[20] M. Julien, SPAMS toolbox, ⟨http://spams-devel.gforge.inria.fr/⟩.

[21] P.R. Foundation, A freeware multispectral image data analysis system, ⟨https://engineering.purdue.edu/biehl/MultiSpec/hyperspectral.html⟩, [Online;(accessed 22-July-2014) (2014).

[22] J.A. Richards, X. Jia, Remote Sensing Digital Image Analysis: an Introduction, Springer-Verlag, New York, 2006.