REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
	New Reprint	-

4. TITLE AND SUBTITLE

Task-Driven Dictionary Learning for HyperspectralImage Classification with Structured SparsityConstraints

5a. CONTRACT NUMBER	W911NF-11-1-0245
5b. GRANT NUMBER	
5c. PROGRAM ELEMENT NUMBER	611102
5d. PROJECT NUMBER	
5e. TASK NUMBER	
5f. WORK UNIT NUMBER	

6. AUTHORS

Xiaoxia Sun, Nasser M. Nasrabadi, Trac D. Tran

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES

Johns Hopkins University
Physics & Astronomy
3400 North Charles Street
Baltimore, MD 21218 -2685

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES)

U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
60291-MA.20

12. DISTRIBUTION AVAILIBILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT

Sparse representation models a signal as a linear combination of a small number of dictionary atoms. As a generative model, it requires the dictionary to be highly redundant in order to ensure both a stable high sparsity level and a low reconstruction error for the signal. However, in practice, this requirement is usually impaired by the lack of labelled

15. SUBJECT TERMS

Sparse representation, supervised dictionarylearning, task-driven dictionary learning, joint sparsity, Laplaciansparsity, hyperspectral imagery classification

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Trac Tran
UU	UU	UU	UU		19b. TELEPHONE NUMBER 410-516-7416

Report Title

Task-Driven Dictionary Learning for HyperspectralImage Classification with Structured SparsityConstraints

ABSTRACT

Sparse representation models a signal as a linear combination of a small number of dictionary atoms. As a generative model, it requires the dictionary to be highly redundant in order to ensure both a stable high sparsity level and a low reconstruction error for the signal. However, in practice, this requirement is usually impaired by the lack of labelled training samples. Fortunately, previous research has shown that the requirement for a redundant dictionary can be less rigorous if simultaneous sparse approximation is employed, which can be carried out by enforcing various structured sparsity constraints on the sparse codes of the neighboring pixels. In addition, numerous works have shown that applying a variety of dictionary learning methods for the sparse representation model can also improve the classification performance. In this paper, we highlight the task-driven dictionary learning algorithm, which is a general framework for the supervised dictionary learning method. We propose to enforce structured sparsity priors on the task-driven dictionary learning method in order to improve the performance of the hyperspectral classification. Our approach is able to benefit from both the advantages of the simultaneous sparse representation and those of the supervised dictionary learning. We enforce two different structured sparsity priors, the joint and Laplacian sparsity, on the task-driven dictionary learning method and provide the details of the corresponding optimization algorithms. Experiments on numerous popular hyperspectral images demonstrate that the classification performance of our approach is superior to sparse representation classifier with structured priors or the task-driven dictionary learning method.

REPORT DOCUMENTATION PAGE (SF298)
(Continuation Sheet)

Continuation for Block 13

ARO Report Number 60291.20-MA
Task-Driven Dictionary Learning for Hyperspect...

Block 13: Supplementary Note

Approved for public release; distribution is unlimited.

Task-Driven Dictionary Learning for Hyperspectral Image Classification with Structured Sparsity Constraints

Xiaoxia Sun, Nasser M. Nasrabadi, *Fellow, IEEE,* and Trac D. Tran, *Fellow, IEEE*

***Abstract*—Sparse representation models a signal as a linear combination of a small number of dictionary atoms. As a generative model, it requires the dictionary to be highly redundant in order to ensure both a stable high sparsity level and a low reconstruction error for the signal. However, in practice, this requirement is usually impaired by the lack of labelled training samples. Fortunately, previous research has shown that the requirement for a redundant dictionary can be less rigorous if simultaneous sparse approximation is employed, which can be carried out by enforcing various structured sparsity constraints on the sparse codes of the neighboring pixels. In addition, numerous works have shown that applying a variety of dictionary learning methods for the sparse representation model can also improve the classification performance. In this paper, we highlight the task-driven dictionary learning algorithm, which is a general framework for the supervised dictionary learning method. We propose to enforce structured sparsity priors on the task-driven dictionary learning method in order to improve the performance of the hyperspectral classification. Our approach is able to benefit from both the advantages of the simultaneous sparse representation and those of the supervised dictionary learning. We enforce two different structured sparsity priors, the joint and Laplacian sparsity, on the task-driven dictionary learning method and provide the details of the corresponding optimization algorithms. Experiments on numerous popular hyperspectral images demonstrate that the classification performance of our approach is superior to sparse representation classifier with structured priors or the task-driven dictionary learning method.**

***Index Terms*—Sparse representation, supervised dictionary learning, task-driven dictionary learning, joint sparsity, Laplacian sparsity, hyperspectral imagery classification**

I. Introduction

CLASSIFICATION on Hyperspectral Imagery (HSI) is becoming increasingly popular in remote sensing. Notable applications include military aerial surveillance [1]–[3], mineral identification and material defects detection [4]. However, numerous difficulties impede the improvement of HSI classification performance. For instance, the high dimensionality of HSI pixels introduce the problem of the 'curse of dimensionality' [5], and the classifier is always confronted with the overfitting problem due to the small number of labelled samples. Additionally, most HSI pixels are indiscriminative since they are undesirably highly coherent [6]. In the past few decades, numerous classification techniques, such as SVM [7], k-nearest-neighbor classifier [8], multimodel logistic regression [9] and neural network [10], have been proposed to alleviate some of these problems to achieve an acceptable performance for HSI classification.

X.Sun and T. D. Tran are with the Department of Electrical and Computer Engineering, The Johns Hopkins University, Baltimore, MD 21218 USA (e-mail: xsun9@jhu.edu; trac@jhu.edu). This work has been partially supported by NSF under Grants CCF-1117545, ARO under Grants 60219-MA, and ONR under grant N000141210765.

N. M. Nasrabadi is with U.S. Army Research Laboratory, Adelphi, MD 20783 USA (e-mail: nnasraba@arl.army.mil).

A. Sparse Representation for HSI classification

More recently, researchers have focused attention on describing the high dimensional data as a sparse linear combination of dictionary atoms. Sparse representation classifier (SRC) was proposed in [11] and has been successfully applied to a wide variety of applications, such as face recognition [11], visual tracking [12], speech recognition [13] and aerial image detection [14]. SRC has also been applied to HSI classification by Chen *et. al.* [15], where a dictionary was constructed by stacking all the labelled samples. Success of SRC requires that the high dimensional data belonging to the same class to lie in a low dimensional subspace. The outstanding classification performance is due to the robustness of sparse recovery, which is largely provided by the high redundancy and low coherency of the dictionary atoms. A low reconstruction error and a high sparsity level can be achieved if the designed dictionary satisfies the above properties. Unfortunately, in practice, the HSI dictionary usually does not have the above properties due to the small number of bluehighly correlated labelled training samples [6].

Due to these undesired properties of the HSI dictionary, the sparse recovery can become unstable and unpredictable such that even pixels belonging to the same class can have totally different sparse codes. The problem induced by the high-coherency of the dictionary atoms, which can be alleviated through decreasing the variation between the sparse codes of the hyperspectral pixels that belong to the same class. In HSI, pixels that are spatially close to each other usually have similar spectral features and belong to the same class. Previous research has shown that the sparse codes of neighboring pixels can become similar by enforcing a structured sparsity constraint (prior). The simultaneous sparse recovery is analytically guaranteed to achieve a sparser solution and a lower reconstruction error with a smaller dictionary [16]. A variety of structured sparsity priors are proposed in the literature [17] that are capable of generating different desired sparsity patterns for the sparse codes of neighboring pixels.

The joint sparsity prior [15] assumes that the features of all the neighboring pixels lie in the same low dimensional subspace and all the corresponding sparse codes share the same set of dictionary atoms. Therefore, the sparse codes have a row sparsity pattern, where only a few rows of the sparse codes are nonzero [18], [19]. The collaborative group sparsity prior [20] enforces the coefficients to have a group-wise sparsity pattern, where the coefficients within each active group are dense. The collaborative hierarchical sparsity prior [21] enforces the sparse codes to be not only group-wise sparse, but also sparse within each active group. The low rank prior [22] assumes that the neighboring pixels are linearly dependent. It does not necessary lead the coefficients to be sparse, which is detrimental for a good classification. However, the low rank group prior proposed in [17] is able to enforce both a group sparsity prior and a low rank prior on the sparse codes by forcing the same group of dictionary atoms to be active if and only if the corresponding neighboring pixels are linearly dependent. The Laplacian sparsity prior [23] uses a Laplacian matrix to describe the degree of similarity between the neighboring pixels. The neighboring pixels that have less spectral features in common are less encouraged to have a similar sparse codes. It has been shown that all the structured sparsity priors are capable of obtaining a smoother classification map and improving the classification performance [17].

B. Dictionary Learning for Sparse Representation

In the classical SRC, the dictionary is constructed by stacking all the training samples. The sparse recovery can be computationally burdensome when the training set is large. Besides, the dictionary constructed in this manner can neither be optimal for reconstruction purposes nor for classification of signals. Previous literature have shown that a dictionary can be trained to have a better representation of the dataset. Unsupervised dictionary learning methods, such as the method of optimal direction (MOD) [24], K-SVD [25] and online dictionary learning [26], are able to improve the signal restoration performance of numerous applications, such as compressive sensing, signal denoising and image inpainting.

However, the unsupervised dictionary learning method is not suitable for solving classification problems since a lower reconstruction error does not necessarily lead to a better classification performance. In fact, it is observed that the dictionary can have an improved classification result by sacrificing some signal reconstruction performance [27]. Therefore, supervised dictionary learning methods [28] are proposed to improve the classification result. Unlike the unsupervised dictionary learning, which only trains the dictionary by pursuing a lower signal reconstruction error, the supervised learning is able to directly improve the classification performance by optimizing both the dictionary and classifier's parameter simultaneously. The discriminative dictionary learning in [29] minimizes the classification error of SRC by minimizing the reconstruction error contributed by the atoms from the correct class and maximizing the error from the remaining classes. The incoherent dictionary learning in [30] uses SRC as the classifier and tries to eliminate the atoms shared by pixels from different classes. It increases the discriminability of the sparse codes by decreasing the coherency of the atoms from different classes. The label consistent K-SVD (LC-KSVD) [31] optimizes the dictionary and classifier's parameter by minimizing the summation of reconstruction and classification errors. It combines the dictionary and classifier's parameter into a single parameter space, which makes it possible for the optimization procedure to be much simpler than those used in classical SRC. However, a desired and accurate solution is not guaranteed [32] because the cost function can be minimized by decreasing the reconstruction error while the classification error is increased. A bilevel optimization formulation would be more appropriate [33], where the update of the dictionary is driven by the minimization of the classification error. The task-driven dictionary learning (TDDL) [27] exploits this idea with theoretical proof and demonstrates a superior performance. The supervised translation-invariant sparse coding, which uses the same scheme as TDDL, is developed independently by [34]. It is a more general framework that can be applied not only to classification, but also nonlinear image mapping, digital art authentication and compressive sensing. More recently, the group sparsity prior is enforced on a single measurement and the corresponding TDDL optimization algorithm is developed in [35] in order to improve the performance of region tagging.

C. Contributions

In this paper, we propose a novel method that enforces the joint or Laplacian sparsity prior on the sparse recovery stage of TDDL. The existing dictionary learning methods have only been developed for reconstructing or classifying a single measurement. Therefore, it is advantageous to incorporate structured sparsity priors into the supervised dictionary learning in order to achieve a better performance. This paper makes the following contributions:

- We propose a new dictionary learning algorithm for TDDL with joint or Laplacian sparsity in order to exploit the spatial-spectral information of HSI neighboring pixels.
- We show experimentally that the proposed dictionary learning methods have a significantly better performance than SRC even when the dictionary is highly compact.
- We also describe an optimization algorithm for solving the Laplacian sparsity recovery problem. The proposed optimization method is much faster than the modified feature sign search used in [23].

The remainder of the paper is organized as follows. In Section II, a brief review of TDDL is given. In Section III, we propose a modified TDDL algorithm with the joint sparsity prior. TDDL with the Laplacian prior and a new algorithm for recovering the Laplacian sparse problem are stated in Section IV. In Section V, we show that our method is superior to other HSI classification methods through experimental results on several HSI images. Finally, we provide our conclusion in Section VI.

II. Task-driven Dictionary Learning

In TDDL [27], signals are represented by their sparse codes, which are then fed into a linear regression or logistic regression. Consider a pair of training samples $(\mathbf{x}, \mathbf{y})$, where $\mathbf{x} \in \mathbb{R}^M$ is the HSI pixel, M is the number of spectral bands, and $\mathbf{y} \in \mathbb{R}^K$ is a binary vector representation of the label of the sample $\mathbf{x}$. K is the maximum class index. Pixel $\mathbf{x}$ can be represented by a sparse coefficient vector $\boldsymbol{\alpha}(\mathbf{D}, \mathbf{x}) \in \mathbb{R}^N$ with respect to some dictionary $\mathbf{D} \in \mathbb{R}^{M \times N}$ consisting of N atoms by solving the optimization

$$\boldsymbol{\alpha}(\mathbf{D}, \mathbf{x}) = \arg\min_{\mathbf{z}} \|\mathbf{x} - \mathbf{D}\mathbf{z}\|_2^2 + \lambda\|\mathbf{z}\|_1 + \frac{\epsilon}{2}\|\mathbf{z}\|_2^2, \quad (1)$$

,where λ and ϵ are the regularization parameters. λ controls the sparsity level of the coefficients $\boldsymbol{\alpha}$. In our experiments, we set ϵ to 0 since it does not affect the convergence of the algorithm and always gives satisfactory results.

To optimize the dictionary, TDDL first defines a convex function $\mathcal{L}(\mathbf{D}, \mathbf{W}, \{\mathbf{x}_i\}_{i=1}^S)$ to describe the classification risk in terms of the dictionary atoms, sparse coefficients and the classifier's parameter $\mathbf{W}$. The function is then minimized as follows

$$\min_{\mathbf{D},\mathbf{W}} \mathcal{L}(\mathbf{D}, \mathbf{W}, \{\mathbf{x}_i\}_{i=1}^S) = \min_{\mathbf{D},\mathbf{W}} f(\mathbf{D}, \mathbf{W}, \{\mathbf{x}_i\}_{i=1}^S) + \frac{\mu}{2}\|\mathbf{W}\|_F^2, \quad (2)$$

where $\mu > 0$ is a classifier regularization parameter to avoid overfitting of the classifier [36]. The convex function f is defined as

$$f(\mathbf{D}, \mathbf{W}, \{\mathbf{x}_i\}_{i=1}^S) \triangleq \frac{1}{S}\sum_{i=1}^{S} \mathcal{J}(\mathbf{y}_i, \mathbf{W}, \boldsymbol{\alpha}_i(\mathbf{D}, \mathbf{x}_i)), \quad (3)$$

where S is the total number of training samples and $\mathcal{L}(\mathbf{y}_i, \mathbf{W}, \boldsymbol{\alpha}_i(\mathbf{D}, \mathbf{x}_i))$ is the classification error for a training pair $(\mathbf{x}_i, \mathbf{y}_i)$ which is measured by a linear regression, i.e. $\mathcal{J}(\mathbf{y}_i, \mathbf{W}, \boldsymbol{\alpha}_i(\mathbf{D}, \mathbf{x}_i)) = \frac{1}{2}\|\mathbf{y}_i - \mathbf{W}\boldsymbol{\alpha}_i\|_2^2$.

In the following part of the section, we omit the subscript i of $\boldsymbol{\alpha}$ for notational simplicity. The dictionary $\mathbf{D}$ and the classifier parameter $\mathbf{W}$ are updated using a stochastic gradient descent algorithm, which has been independently investigated by [27], [34]. The update rules for $\mathbf{D}$ and $\mathbf{W}$ are

$$\begin{cases} \mathbf{D}^{(t+1)} = \mathbf{D}^{(t)} - \rho^{(t)} \cdot \partial\mathcal{L}^{(t)}/\partial\mathbf{D}, \\ \mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} - \rho^{(t)} \cdot \partial\mathcal{L}^{(t)}/\partial\mathbf{W}, \end{cases} \quad (4)$$

where t is the iteration index and ρ is the step size. The equations for updating the classifier parameter $\mathbf{W}$ is straightforward since $\mathcal{L}(\mathbf{y}, \mathbf{W}, \boldsymbol{\alpha}(\mathbf{D}, \mathbf{x}))$ is both smooth and convex with respect to $\mathbf{W}$. We have

$$\frac{\partial\mathcal{L}}{\partial\mathbf{W}} = (\mathbf{W}\boldsymbol{\alpha} - \mathbf{y})\,\boldsymbol{\alpha}^\top + \mu\mathbf{W}. \quad (5)$$

The updating equation for the dictionary can be obtained by applying error backpropagation, where the chain rule is applied

$$\frac{\partial\mathcal{L}}{\partial\mathbf{D}} = \frac{\partial\mathcal{L}}{\partial\boldsymbol{\alpha}}\frac{\partial\boldsymbol{\alpha}}{\partial\mathbf{D}}. \quad (6)$$

The difficulty of acquiring a specific form of the above equation comes from $\partial\boldsymbol{\alpha}/\partial\mathbf{D}$. Since the sparse coefficient $\boldsymbol{\alpha}(\mathbf{D}, \mathbf{x})$ is an implicit function of $\mathbf{D}$, an analytic form of $\boldsymbol{\alpha}$ with respect to $\mathbf{D}$ is not available. Fortunately, the derivative $\partial\boldsymbol{\alpha}/\partial\mathbf{D}$ can still be computed by either applying optimality condition of elastic net [27], [37] or using fixed point differentiation [34], [38].

We now focus on computing the derivative using the fixed point differentiation. As suggested in [38], the gradient of Eq. (1) reaches $\mathbf{0}$ at the optimal point $\hat{\boldsymbol{\alpha}}$

$$\frac{\partial\|\mathbf{x} - \mathbf{D}\boldsymbol{\alpha}\|_2^2}{\partial\boldsymbol{\alpha}}\Big|_{\boldsymbol{\alpha}=\hat{\boldsymbol{\alpha}}} = -\lambda\frac{\partial\|\boldsymbol{\alpha}\|_1}{\partial\boldsymbol{\alpha}}\Big|_{\boldsymbol{\alpha}=\hat{\boldsymbol{\alpha}}}. \quad (7)$$

Expanding Eq. (7), we have

$$2\mathbf{D}^\top(\mathbf{x} - \mathbf{D}\boldsymbol{\alpha})\Big|_{\boldsymbol{\alpha}=\hat{\boldsymbol{\alpha}}} = \lambda \cdot sign(\boldsymbol{\alpha})\Big|_{\boldsymbol{\alpha}=\hat{\boldsymbol{\alpha}}}. \quad (8)$$

In order to evaluate $\partial\boldsymbol{\alpha}/\partial\mathbf{D}$, the derivative of Eq. (8) with respect to each element D_{mn} of the dictionary is required. Since the differentiation of the $sign$ function is not well defined at zero points, we can only compute the derivative of Eq. (8) at fixed points when $\boldsymbol{\alpha}_{[n]} \neq 0$ [34]

$$\frac{\partial\boldsymbol{\alpha}_\Lambda}{\partial D_{mn}} = (\mathbf{D}_\Lambda^\top\mathbf{D}_\Lambda)^{-1}\left(\frac{\partial\mathbf{D}_\Lambda^\top\mathbf{x}}{\partial D_{mn}} - \frac{\partial\mathbf{D}_\Lambda^\top\mathbf{D}_\Lambda}{\partial D_{mn}}\boldsymbol{\alpha}_\Lambda\right) \text{ and } \frac{\partial\boldsymbol{\alpha}_{\Lambda^C}}{\partial D_{mn}} = \mathbf{0}, \quad (9)$$

where Λ and Λ^c are the indices of the active and inactive set of $\boldsymbol{\alpha}$ respectively. $D_{mn} \in \mathbb{R}$ is the (m, n) element of $\mathbf{D}$. $(\mathbf{D}_\Lambda^\top\mathbf{D}_\Lambda)^{-1}$ is always invertible since the number of active atoms $|\Lambda|$ is always much smaller than the feature dimension M.

III. TDDL with joint sparsity prior

We now extend TDDL by using a joint sparsity (JS) prior (TDDL-JS). The joint sparsity prior [18], [19] enforces the sparse coefficients of the test pixel and its neighboring pixels within the neighborhood window to have row sparsity pattern, where all pixels are represented by the same atoms in the dictionary so that only few rows of the sparse coefficients matrix are nonzero. The joint sparse recovery can be solved by the following Lasso problem

$$\mathbf{A} = \arg\min_{\mathbf{Z}} \|\mathbf{X} - \mathbf{D}\mathbf{Z}\|_F^2 + \lambda\|\mathbf{Z}\|_{1,2}, \quad (10)$$

where $\mathbf{A}, \mathbf{Z} \in \mathbb{R}^{N \times P}$ are sparse coefficient matrices and $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_P] \in \mathbb{R}^{M \times P}$ represents all the pixels within a neighborhood window centered on a test (center) pixel $\mathbf{x}_c$. Define the label of the center pixel as $\mathbf{y}_c$. P is the total number of pixels within the neighborhood window. $\|\mathbf{Z}\|_{1,2} = \sum_{i=1}^{P}\|\mathbf{Z}_i\|_2$ is the $\ell_{1,2}$-norm of $\mathbf{Z}$. $\mathbf{Z}_i \in \mathbb{R}^{1 \times P}$ is the i^{th} row of $\mathbf{Z}$. Many sparse recovery techniques are able to solve Eq. (10), such as the Alternating Direction Method of Multipliers [39], Sparse Reconstruction by Separable Approximation (SpaRSA) [40] and Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) [41].

Once the sparse code $\mathbf{A}$ is obtained, the sparse codes $\boldsymbol{\alpha}_c$ of the center pixel $\mathbf{x}_c$ is projected on each of the K decision planes of the classifier. The plane with the largest projection indicates the class that the center pixel $\mathbf{x}_c$ belongs to,

$$\text{identity}(\mathbf{x}_c) = \arg\max_k \hat{\mathbf{y}}_k = \arg\max_k (\mathbf{W}\boldsymbol{\alpha}_c)_k, \quad (11)$$

where $\boldsymbol{\alpha}_c \in \mathbb{R}^N$ is the sparse coefficients of the center pixel. In the training stage, it is expected that the projection of the decision plane corresponding to the class of the center pixel should be increased while other planes should be orthogonal to $\boldsymbol{\alpha}_c$. Therefore, given the training data $(\mathbf{X}, \mathbf{y}_c)$, the classification error for the center pixel $\mathbf{x}_c$ is defined as

$$\mathcal{L}(\mathbf{y}_c, \mathbf{W}, \boldsymbol{\alpha}_c(\mathbf{D}, \mathbf{X})) = \|\mathbf{y}_c - \mathbf{W}\boldsymbol{\alpha}_c\|_2^2 + \frac{\mu}{2}\|\mathbf{W}\|_F^2, \quad (12)$$

In order to update the dictionary $\mathbf{D}$, we need to apply a chain rule similar to the one in Eq. (6):

$$\frac{\partial \mathcal{L}}{\partial \mathbf{D}} = \frac{\partial \mathcal{L}}{\partial \mathbf{A}} \frac{\partial \mathbf{A}}{\partial \mathbf{D}}. \quad (13)$$

Now we focus on the difficult part $\frac{\partial \mathbf{A}}{\partial \mathbf{D}}$ of Eq. (13). Employing the fixed point differentiation on Eq. (10), we have

$$\frac{\partial \|\mathbf{X} - \mathbf{D}\mathbf{A}\|_F^2}{\partial \mathbf{A}}\Big|_{\mathbf{A}=\hat{\mathbf{A}}} = -\lambda \frac{\partial \|\mathbf{A}\|_{1,2}}{\partial \mathbf{A}}\Big|_{\mathbf{A}=\hat{\mathbf{A}}}. \quad (14)$$

In the following part of this section, we omit the fixed point notation. Eq. (14) is only differentiable when $\|\mathbf{A}_i\|_2 \neq \mathbf{0}$, where $\mathbf{A}_i$ denotes the i^{th} row of $\mathbf{A}$. At points where $\|\mathbf{A}_i\|_2 = \mathbf{0}$, the derivative is not well defined, so we set $\frac{\partial \|\mathbf{A}_i\|_2}{\partial \mathbf{A}_i} = \mathbf{0}$. Denote $\tilde{\mathbf{A}} = \mathbf{A}_\Lambda \in \mathbb{R}^{N_\Lambda \times P}$, where Λ is the active set such that $\Lambda = \{i : \|\mathbf{A}_i\|_2 \neq 0, i \in \{1, \ldots, N\}\}$, $N_\Lambda = |\Lambda|$, $\mathbf{A}_\Lambda$ is composed of active rows of $\mathbf{A}$, and $\tilde{\mathbf{D}}$ is the active atoms of $\mathbf{D}$. Expanding the derivative of Eq. (14) on both sides on the feasible points,

$$\tilde{\mathbf{D}}^\top \left(\mathbf{X} - \tilde{\mathbf{D}}\tilde{\mathbf{A}}\right) = \lambda \left[\frac{\tilde{\mathbf{A}}_1^\top}{\|\tilde{\mathbf{A}}_1\|_2}, \ldots, \frac{\tilde{\mathbf{A}}_{N_\Lambda}^\top}{\|\tilde{\mathbf{A}}_{N_\Lambda}\|_2}\right]^\top. \quad (15)$$

Computing the derivative of Eq. (15) with respect to D_{mn} and transposing both sides

$$\frac{\partial \left\{(\mathbf{X} - \mathbf{D}\mathbf{A})^\top \tilde{\mathbf{D}}\right\}}{\partial D_{mn}} = \lambda \left[\boldsymbol{\Gamma}_1 \frac{\partial \tilde{\mathbf{A}}_1^\top}{\partial D_{mn}}, \ldots, \boldsymbol{\Gamma}_{N_\Lambda} \frac{\partial \tilde{\mathbf{A}}_{N_\Lambda}^\top}{\partial D_{mn}}\right], \quad (16)$$

where $\boldsymbol{\Gamma}_i = \frac{\mathbf{I}_P}{\|\tilde{\mathbf{A}}_i\|_2} - \frac{\tilde{\mathbf{A}}_i^\top \tilde{\mathbf{A}}_i}{\|\tilde{\mathbf{A}}_i\|_2^3}$, $i = 1, \ldots, N_\Lambda$. By vectorizing Eq. (16), we have

$$vec\left(\frac{\partial \mathbf{X}^\top \tilde{\mathbf{D}}}{\partial D_{mn}} - \tilde{\mathbf{A}}^\top \frac{\partial \tilde{\mathbf{D}}^\top \tilde{\mathbf{D}}}{\partial D_{mn}} - \frac{\partial \tilde{\mathbf{A}}^\top}{\partial D_{mn}} \tilde{\mathbf{D}}^\top \tilde{\mathbf{D}}\right) = \lambda \cdot \boldsymbol{\Gamma} vec\left(\frac{\partial \tilde{\mathbf{A}}^\top}{\partial D_{mn}}\right), \quad (17)$$

where $\boldsymbol{\Gamma} = \boldsymbol{\Gamma}_1 \oplus \cdots \oplus \boldsymbol{\Gamma}_{N_\Lambda}$. From Eq. (17), we reach the vectorization form of the derivative of $\tilde{\mathbf{A}}$ with respect to D_{mn}, given as

$$vec\left(\frac{\partial \tilde{\mathbf{A}}^\top}{\partial D_{mn}}\right) = \left(\tilde{\mathbf{D}}^\top \tilde{\mathbf{D}} \otimes \mathbf{I}_P + \lambda \boldsymbol{\Gamma}\right)^{-1} vec\left(\tilde{\mathbf{A}}^\top \frac{\partial \tilde{\mathbf{D}}^\top \tilde{\mathbf{D}}}{\partial D_{mn}} + \frac{\partial \mathbf{X}^\top \tilde{\mathbf{D}}}{\partial D_{mn}}\right). \quad (18)$$

Now we can update the dictionary element-wise using Eq. (18). In order to reach a more concise form for updating the dictionary, we perform algebraic transformations on Eq. (13) and Eq. (18), which are illustrated in Appendix VII. We illustrate the overall optimization for TDDL-JS in Algorithm 1. It should be noted that in the Algorithm 1, we define $\hat{\mathbf{A}} = [\mathbf{0}, \ldots, \boldsymbol{\alpha}_c, \ldots, \mathbf{0}] \in \mathbb{R}^{N \times P}$ and $\hat{\mathbf{Y}} = [\mathbf{0}, \ldots, \mathbf{y}, \ldots, \mathbf{0}] \in \mathbb{R}^{K \times P}$.

Algorithm 1 Stochastic gradient descent algorithm for task-driven dictionary learning with joint sparsity prior

Require: Initial dictionary $\mathbf{D}$ and classifier $\mathbf{W}$. Parameter λ, ρ and t_0.

1: **for** $t = 1$ to T **do**

2: Draw one sample $(\mathbf{X}, \mathbf{y}_c)$ from training set.

3: Find sparse sparse code $\mathbf{A}$ according to Eq. (10).

4: Find the active set Λ and define $N_\Lambda = |\Lambda|$

$$\Lambda \leftarrow \{i : \|\mathbf{A}_i\|_2 \neq 0, i \in \{1, \ldots, N\}\},$$

where $\mathbf{A}_i$ is the i^{th} row of $\mathbf{A}$.

5: Compute $\boldsymbol{\Gamma} \in \mathbb{R}^{N_\Lambda P \times N_\Lambda P}$

$$\boldsymbol{\Gamma} = \boldsymbol{\Gamma}_1 \oplus \cdots \oplus \boldsymbol{\Gamma}_{N_\Lambda},$$
$$\boldsymbol{\Gamma}_i = \frac{\mathbf{I}_P}{\|\tilde{\mathbf{A}}_i\|_2} - \frac{\tilde{\mathbf{A}}_i \tilde{\mathbf{A}}_i^\top}{\|\tilde{\mathbf{A}}_i\|_2^3}, i = 1, \ldots, N_\Lambda,$$

where $\oplus$ is the direct sum of matrices.

6: Compute $\boldsymbol{\gamma} \in \mathbb{R}^{N_\Lambda P}$

$$\boldsymbol{\gamma} = (\tilde{\mathbf{D}}^\top \tilde{\mathbf{D}} \otimes \mathbf{I}_P + \lambda \boldsymbol{\Gamma})^{-\top} vec((\mathbf{W}\hat{\mathbf{A}} - \hat{\mathbf{Y}})^\top \tilde{\mathbf{W}}).$$

where $vec(\cdot)$ and $\tilde{\mathbf{W}}$ denote the vectorization operator and Λ columns of $\mathbf{W}$ respectively.

7: Let $\boldsymbol{\beta} \in \mathbb{R}^{N \times P}$. Set $\boldsymbol{\beta}_{\Lambda^c} = \mathbf{0}$ and construct $\boldsymbol{\beta}_\Lambda \in \mathbb{R}^{N_\Lambda \times P}$ that satisfies

$$vec\left(\boldsymbol{\beta}_\Lambda^\top\right) = \boldsymbol{\gamma}.$$

8: Choose the learning rate $\rho_t \leftarrow \min(\rho, \rho \frac{t_0}{t})$.

9: Update the parameters by gradient projection step

$$\mathbf{W} \leftarrow \mathbf{W} - \rho_t\big((\mathbf{W}\boldsymbol{\alpha}_c - \mathbf{y})\boldsymbol{\alpha}_c^\top + \mu\mathbf{W}\big),$$
$$\mathbf{D} \leftarrow \mathbf{D} - \rho_t(-\mathbf{D}\boldsymbol{\beta}\mathbf{A}^\top + (\mathbf{X} - \mathbf{D}\mathbf{A})\boldsymbol{\beta}^\top),$$

and normalize every column of $\mathbf{D}^{(t+1)}$ with respect to ℓ_2-norm.

10: **end for**

11: **return** $\mathbf{D}$ and $\mathbf{W}$.

IV. TDDL WITH LAPLACIAN SPARSITY PRIOR

The joint sparsity prior is a relatively stringent constraint on the sparse codes since it assumes that all the neighboring pixels have the same support as the center pixel. The assumption of the joint sparsity prior can easily be violated on non-homogeneous regions, such as a region that contains pixels from different classes. This makes choosing a proper neighborhood window size a difficult problem. When the window size is too large, the sparse codes of the non-homogeneous regions within the window are indiscriminative. On the other hand, the sparse codes are not stable if the window size is chosen to be too small. Ideally, we hope that the performance is insensitive to both the choice of the window size and the topology of the image. To achieve this requirement, we propose to enforce the Laplacian sparsity (LP) prior (TDDL-LP) on the TDDL, where the degree of similarity between neighboring pixels can be utilized to push the sparse codes of the neighboring pixels that belong to the same class to be similar, instead of enforcing all

the neighboring pixels to have a similar sparse codes blindly. The corresponding Lasso problem can be stated as follows

$$\mathbf{A} = \arg\min_{\mathbf{Z}} \|\mathbf{X} - \mathbf{D}\mathbf{Z}\|_2^2 + \lambda\|\mathbf{Z}\|_1 + \gamma \sum_{i,j}^{P} c_{ij}\|\mathbf{Z}^i - \mathbf{Z}^j\|_2^2, \tag{19}$$

where $\mathbf{Z}^i$ and $\mathbf{Z}^j$ denote the i^{th} and j^{th} columns of $\mathbf{Z}$. c_{ij} is a weight whose value is proportional to the spectral similarity of $\mathbf{X}^i$ and $\mathbf{X}^j$, which are the i^{th} and j^{th} columns of $\mathbf{X}$. γ is a regularization parameter.

The Laplacian sparse recovery described by Eq. (19) in [23] is able to discriminate pixels from different classes by defining an appropriate weighting matrix $\mathbf{C} = [c_{ij}] \in \mathbb{R}^{P\times P}$. Additionally, it enforces both the support and the magnitude of sparse coefficients of similar spectral pixels to be similar, whereas the joint sparsity prior enforces sparse coefficients of all the pixels within the neighborhood window to have the same support. Eq. (19) can be reformulated as

$$\mathbf{A} = \arg\min_{\mathbf{Z}} \|\mathbf{X} - \mathbf{D}\mathbf{Z}\|_2^2 + \lambda\|\mathbf{Z}\|_1 + \gamma tr(\mathbf{Z}\mathbf{L}\mathbf{Z}^\top), \tag{20}$$

where $\mathbf{L} = \mathbf{B} - \mathbf{C} \in \mathbb{R}^{P\times P}$ is the Laplacian matrix [42]. $\mathbf{B} = [b_{ij}] \in \mathbb{R}^{P\times P}$ is a diagonal matrix such that $b_{ii} = \sum_j c_{ij}$.

In this paper, we adopt the method of Sparse Reconstruction by Separable Approximation (SpaRSA) [17], [40] to solve the Laplacian sparse coding problem.

A. Sparse Recovery Algorithm

A modified feature sign search [23] is capable of solving the optimization problem (20). It uses coordinate descent to update each column of $\mathbf{A}$ iteratively. Although it gives plausible performance for the SRC-based HSI classification [17], it demands a high computational cost. The SpaRSA-based method can achieve a similar optimal solution of Eq. (20) while being less computational burdensome. Despite the fact that our previous work [17] has shown that the performance of the SRC-based approach for HSI classification can be largely influenced by the choice of specific optimization technique, we found that such influence is reasonably small when employing the dictionary-learning-based approach. Therefore, we use a SpaRSA-based method to solve the sparse recovery for the Laplacian sparsity prior. Although, SpaRSA is originally designed to solve the optimization of single-signal case, it can be easily extended to tackle the problem with multiple signals, such as the collaborative hierarchical Lasso (C-Hilasso) [21].

SpaRSA is able to solve optimization problems that have the following form

$$\min_{\mathbf{A}\in\mathbb{R}^{N\times P}} f(\mathbf{A}) + \lambda\psi(\mathbf{A}), \tag{21}$$

where $f : \mathbb{R}^{N\times P} \to \mathbb{R}$ is a convex and smooth function, $\psi : \mathbb{R}^{N\times P} \to \mathbb{R}$ is a separable regularizer and λ is the regularization parameter. In the particular case of the Laplacian sparse recovery, the regularizer ψ is chosen to be the ℓ_1-norm, i.e. $\psi(\mathbf{A}) = \|\mathbf{A}\|_1$, and the convex function f is set as

$$f(\mathbf{A}) = \|\mathbf{X} - \mathbf{D}\mathbf{A}\|_F^2 + \gamma tr\left(\mathbf{A}\mathbf{L}\mathbf{A}^\top\right). \tag{22}$$

Algorithm 2 Sparse recovery for Laplacian sparsity prior using SpaRSA

Require: Dictionary $\mathbf{D}$, constants $\eta_0 > 0$, $0 < \eta_{\min} < \eta_{\max}$, $\mu > 1$

```
Set t = 0 and A^(0) = 0
repeat
    choose η^(t) ∈ [η_min, η_max]
    compute U^(t) ← A^(t) − (1/η^(t)) ∇f(A^(t)).
    repeat
        A^(t) ← S_{γ/η^(t)} (U^(t)),
        η^(t) ← μ η^(t).
    until stopping criterion is satisfied
    t ← t + 1.
until stopping criterion is satisfied
return The optimal sparse coefficients A*.
```

In order to search the optimal solution of Eq. (21), SpaRSA generates a sequence of iterations $\mathbf{A}^{(t)}$, $t = 1, 2, \ldots$, by solving the following subproblem

$$\mathbf{A}^{(t+1)} \in \arg\min_{\mathbf{Z}\in\mathbb{R}^{N\times P}} \left(\mathbf{Z} - \mathbf{A}^{(t)}\right)^\top \nabla f(\mathbf{A}^{(t)}) + \frac{\eta^{(t)}}{2}\|\mathbf{Z} - \mathbf{A}^{(t)}\|_F^2 + \gamma\psi(\mathbf{Z}), \tag{23}$$

where $\eta^{(t)} > 0$ is a nonnegative scalar such that $\eta^{(t)} = \mu\eta^{(t-1)}$ and $\mu > 1$. The Eq. (23) can be simplified into the following form by eliminating the terms independent of $\mathbf{Z}$

$$\min_{\mathbf{Z}\in\mathbb{R}^{N\times P}} \frac{1}{2}\|\mathbf{Z} - \mathbf{U}^{(t)}\|_F^2 + \frac{\gamma}{\eta^{(t)}}\psi(\mathbf{Z}), \tag{24}$$

where $\mathbf{U}^{(t)} = \mathbf{A}^{(t)} - \frac{1}{\eta^{(t)}}\nabla f(\mathbf{A}^{(t)})$. The optimization problem in Eq. (24) is separable element-wise, which can be reformulated into

$$\min_{A_{ij}} \frac{1}{2}(z_{ij} - u_{ij}^{(t)})^2 + \frac{\lambda}{\eta^{(t)}}\psi_{ij}(\mathbf{Z}), \forall i = 1, \ldots, N \text{ and } j = 1, \ldots, P. \tag{25}$$

The problem in Eq. (25) has a unique solution and can be solved by the well-known soft thresholding operator $S(\cdot)$

$$z_{ij}^* = S_{\frac{\gamma}{\eta^{(t)}}}\left(u_{ij}^{(t)}\right) = sign(u_{ij}^{(t)})\max\{0, |u_{ij}| - \frac{\lambda}{\eta^{(t)}}\}. \tag{26}$$

Comparing with the algorithm proposed in [23], which is based on the coordinate descent, Laplacian sparse recovery using SpaRSA is more computationally efficient since it is able to cheaply search for a better descent direction $\nabla f(\mathbf{A})$. The corresponding optimization is stated in Algorithm 2.

B. Dictionary Update

In order to adjust the dictionary, we now follow Eq. (13) to derive $\frac{\partial\mathbf{A}}{\partial\mathbf{D}}$ using the fixed point differentiation. Applying differentiation on Eq. (19) on the fixed point $\hat{\mathbf{A}}$

$$\frac{\partial\|\mathbf{X} - \mathbf{D}\mathbf{A}\|_F^2 + \gamma tr\left(\mathbf{A}\mathbf{L}\mathbf{A}^\top\right)}{\partial\mathbf{A}}\Big|_{\mathbf{A}=\hat{\mathbf{A}}} = -\lambda\frac{\partial\|\mathbf{A}\|_1}{\partial\mathbf{A}}\Big|_{\mathbf{A}=\hat{\mathbf{A}}}. \tag{27}$$

In the following part, we omit the fixed point notation. By computing the derivation and then applying the vectorization on Eq. (27), we have

$$vec\left(\mathbf{D}^\top(\mathbf{X} - \mathbf{D}\mathbf{A}) - \gamma\mathbf{A}\mathbf{L}\right) = \lambda \cdot vec\left(sign(\mathbf{A})\right). \tag{28}$$

Algorithm 3 Stochastic gradient descent algorithm for task-driven dictionary learning with Laplacian sparsity prior

Require: Initial dictionary $\mathbf{D}$ and classifier $\mathbf{W}$. Parameter λ, ρ and t_0.

1: **for** $t = 1$ to T **do**

2: Draw one sample $(\mathbf{X}, \mathbf{y}_c)$ from training set.

3: Find sparse code $\mathbf{A}$ according to Eq. (10).

4: Find the active set Λ

$$\Lambda \leftarrow \{i : vec(\mathbf{A})_i \neq 0, i \in \{1, \ldots, NP\}\},$$

where $vec(\mathbf{A})_i$ is the i^{th} element of $vec(\mathbf{A})$.

5: Let $\boldsymbol{\beta} \in \mathbb{R}^{N\times P}$. Set $vec(\boldsymbol{\beta})_{\Lambda^C} = \mathbf{0}$ and compute $vec(\boldsymbol{\beta})_\Lambda$

$$vec(\boldsymbol{\beta})_\Lambda = (\mathbf{I}_P \otimes \mathbf{D}^\top\mathbf{D} + \gamma\mathbf{L} \otimes \mathbf{I}_N)^{-1}_{\Lambda,\Lambda} vec(\mathbf{W}^\top(\mathbf{W}\hat{\mathbf{A}} - \hat{\mathbf{Y}}))_\Lambda,$$

and $\otimes$ denotes the Kronecker product.

6: Choose the learning rate $\rho_t \leftarrow \min(\rho, \rho\frac{t_0}{t})$.

7: Update the parameters by gradient projection step

$$\mathbf{W} \leftarrow \mathbf{W} - \rho_t\big((\mathbf{W}\boldsymbol{\alpha}_c - \mathbf{y})\boldsymbol{\alpha}_c^\top + \mu\mathbf{W}\big),$$
$$\mathbf{D} \leftarrow \mathbf{D} - \rho_t(-\mathbf{D}\boldsymbol{\beta}\mathbf{A}^\top + (\mathbf{X} - \mathbf{D}\mathbf{A})\boldsymbol{\beta}^\top),$$

and normalize every column of $\mathbf{D}^{(t+1)}$ with respect to ℓ_2-norm.

8: **end for**

9: **return** $\mathbf{D}$ and $\mathbf{W}$.

The differentiation $\frac{\partial vec(sign(\mathbf{A}))}{\partial D_{mn}}$ is not well defined on zero points of $vec\left(sign\left(\mathbf{A}\right)\right)$. Similar as in TDDL-JS, we set the i^{th} element $\frac{\partial vec(sign(\mathbf{A}))_i}{\partial D_{mn}} = 0$ when $vec\left(sign\left(\mathbf{A}\right)\right)_i = 0$. Denote the Λ as the index set of nonzero elements of $vec\left(sign\left(\mathbf{A}\right)\right)$. Compute the derivative of Eq. (28) with respect to D_{mn}

$$\frac{\partial\left\{vec\left(\mathbf{D}^\top\left(\mathbf{X} - \mathbf{D}\mathbf{A}\right) - \gamma\mathbf{A}\mathbf{L}\right)_\Lambda\right\}}{\partial D_{mn}} = \mathbf{0}, \tag{29}$$

which leads to

$$vec\left(\frac{\partial\mathbf{D}^\top\mathbf{D}}{\partial D_{mn}}\mathbf{A} - \frac{\partial\mathbf{D}^\top\mathbf{X}}{\partial D_{mn}} + \mathbf{D}^\top\mathbf{D}\frac{\partial\mathbf{A}}{\partial D_{mn}} + \gamma\frac{\partial\mathbf{A}}{\partial D_{mn}}\mathbf{L}\right)_\Lambda = \mathbf{0}. \tag{30}$$

Now we reach the desired gradient

$$vec\left(\frac{\partial\mathbf{A}}{\partial D_{mn}}\right)_\Lambda = \left(\mathbf{I}_P \otimes \mathbf{D}^\top\mathbf{D} + \gamma\mathbf{L} \otimes \mathbf{I}_N\right)^{-1}_{\Lambda,\Lambda} vec\left(\frac{\partial\tilde{\mathbf{D}}^\top\tilde{\mathbf{D}}}{\partial D_{mn}}\tilde{\mathbf{A}} + \frac{\partial\tilde{\mathbf{D}}^\top\mathbf{X}}{\partial D_{mn}}\right)_\Lambda. \tag{31}$$

By applying algebraic simplification to Eq. (31), which is shown in Appendix VII, we reach the optimzation for TDDL-LP as stated in the Algorithm 3. It should be noted that $\hat{\mathbf{A}}$ and $\hat{\mathbf{Y}}$ have the same definitions as those in Algorithm 1.

V. EXPERIMENTS

A. Experiment Setup

Cross-validation to obtain the optimal values for all parameters, including $\lambda, \epsilon, \gamma$ (sparse coding regularization parameters), μ (regularization parameter for the classifier), ρ_0 (initial step size), N (dictionary size) and P (number of neighboring pixels), would introduce significant computational cost. Instead, we search for the optimal values for the above parameters according to the following procedure.

- The candidate dictionary sizes are from 5 to 10 atoms per class. The choice of dictionary size depends on the classification performance and computational cost. In our experiment, we set the dictionary size to be 5 atoms per class.
- Searching for the optimal window size and the regularization parameters would be cumbersome. Empirically, we found that the optimal regularization parameters are less likely to be affected by the choice of the window size. Therefore, for each image, we fix the window size to be 3×3 in order to save computational resource during the search of the optimal regularization parameters. Candidate regularization parameters are $\{10^{-3}, 10^{-2}, 10^{-1}\}$.
- The possible candidate window sizes are 3×3, 5×5, 7×7 and 9×9. We search for the optimal window size for each image after finding the optimal regularization parameters.

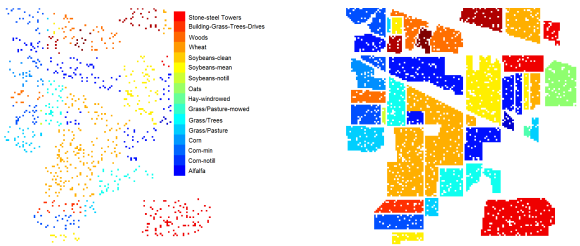
TABLE I: Parameters Used in the Paper

Structured Priors	λ	γ	ρ
ℓ_1	10^{-2}		10^{-2}
JS	10^{-2}		10^{-3}
LP	10^{-2}	10^{-3}	10^{-1}

Computing the gradient for a single training sample at each iteration of Algorithm 1 or 3 will make the algorithm converge very slowly. Therefore, following the previous work [26], [27], we implement the two proposed algorithms with the mini-batch method, where the gradients of multiple training samples are computed in each iteration. For the unsupervised learning methods, the batch size is set to 200. For the supervised learning methods, the batch size is set to 100 and $t_0 = T/10$. We search the optimal regularization parameters for each image and found that their optimal values are coincidentally the same. The reason could be due to our choice of a large interval for the search grid. The regularization parameters used in our paper are shown in Table I. We set $\mu = 10^{-4}$. As a standard procedure, we evaluate the classification performance on HSI image using the overall accuracy (OA), average accuracy (AA) and kappa coefficient (κ). The classification methods that are tested and compared are SVM, SRC, SRC with joint sparsity prior (SRC-JS), SRC with Laplacian sparsity prior (SRC-LP), unsupervised dictionary learning (ODL), unsupervised dictionary learning with joint or Laplacian sparsity prior (ODL-JS, ODL-LP), TDDL, TDDL-JS and TDDL-LP. During the testing stage, all training pixels are excluded from the HSI image, which means there may be some 'holes' (training pixels deleted) inside a neighborhood window. This is reasonable since we do not want the classification results to be affected by the spatial distribution of the labelled samples. We use SPAMS toolbox [43] to perform the joint sparse recovery via the Fast Iterative Shrinkage-Thresholding Algorithm [41]. The sparse recovery for SRC-based methods are performed

via the Alternating Direction Method of Multipliers [39]. The modified SpaRSA shown in Algorithm 2 is used to solve the Laplacian sparse recovery problem.

For the unsupervised dictionary learning methods, the dictionary is initialized by randomly choosing a subset of the training pixels from each class and updated using the online dictionary learning (ODL) procedure in [26]. The classifier's parameter are then obtained by using a multi-class linear regression. For the supervised dictionary learning methods, the dictionary and classifier's parameter are initialized by the training results of ODL for the unsupervised method.



(a) (b)

Fig. 1: (a) Training sets and (b) test sets of the Indian Pine image.

TABLE II: Number of training and test samples for the Indian Pine image

Class #	Name	Train	Test
1	Alfalfa	6	48
2	Corn-notill	137	1297
3	Corn-min	80	754
4	Corn	23	211
5	Grass/Pasture	48	449
6	Grass/Trees	72	675
7	Grass/Pasture-mowed	3	23
8	Hay-windrowed	47	442
9	Oats	2	18
10	Soybeans-notill	93	875
11	Soybeans-min	235	2233
12	Soybean-clean	59	555
13	Wheat	21	191
14	Woods	124	1170
15	Building-Grass-Trees-Drives	37	343
16	Stone-steel Towers	10	85
Total		997	9369

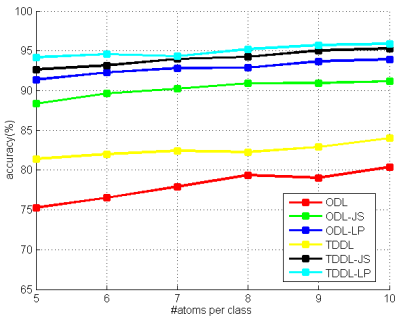


Fig. 2: The result with different dictionary sizes for the Indian Pine image.

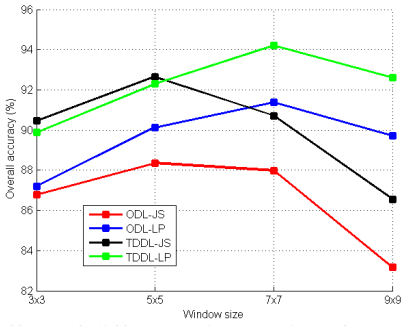


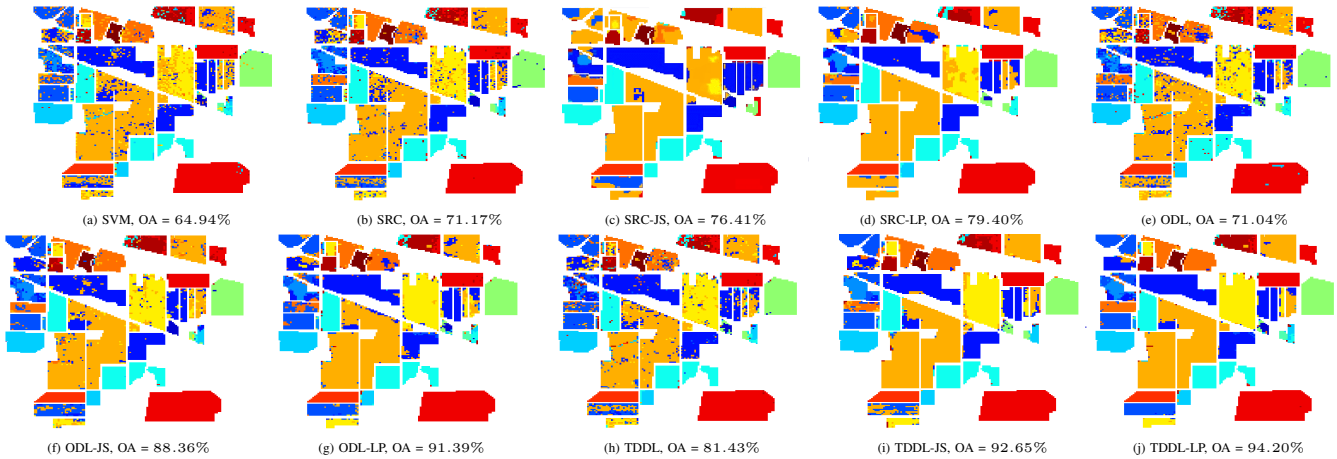
Fig. 3: The effect of different window sizes for the Indian Pine image. The dictionary size is fixed at five atoms per class.

B. Classification on AVIRIS Indian Pine Dataset

We first perform HSI classification on the Indian Pine image, which is generated by Airborne Visible/Infrared Imaging Spectrometer (AVIRIS). Every pixel of the Indian Pine consists of 220 bands ranging from 0.2 to 2.4μm, of which 20 water absorption bands are removed before classification. The spatial dimension of this image is 145×145. The image contains 16 ground-truth classes, most of which are crops, as shown in Table II. We randomly choose 997 pixels (10.64% of all the interested pixels) as the training set and the rest of the interested pixels for testing.

The total iterations of unsupervised and supervised dictionary learning methods are set to 15 and 200 respectively for this image. The classification results with varying dictionary size N are shown in Fig. 2. In most cases, the classification performance increases with the increment in the dictionary size. All methods attain their highest OA when the dictionary size is 10 atoms per class. The OA of ODL-JS, ODL-LP, TDDL-JS and TDDL-LP do not change much when the dictionary size increase from 5 to 10 atoms per class. Therefore, it is reasonable to set the dictionary size to be 5 atoms per class by taking computational cost into account. Fig. 2 also suggests that a plausible performance can be obtained even when the dictionary is very small and not over-complete. The classification performance with respect to the window size is demonstrated in Fig. 3. Using a window size of 5×5, ODL-JS and TDDL-JS achieves the highest OA of 88.36% and 92.65%, respectively. When the window size is set to 7×7, the ODL-LP and TDDL-LP reach their highest OA $= 91.39\%$ and OA $= 94.20\%$, respectively. ODL-JS and TDDL-JS reach better performance when the window size is not larger than 5×5. The TDDL-LP outperforms all other methods when the window size is 7×7 or larger. Since a larger window size has more chances to include non-homogeneous regions, it verifies our argument that the Laplacian sparsity prior works better for classifying pixels lying in the non-homogeneous regions.

Detailed classification results of various methods are shown in Table III and visually displayed in Fig. 4. The OA of ODL-LP reaches 91.39%, which is more than 20% higher than that of ODL and 3% higher than that of ODL-JS. The TDDL-LP has the highest classification accuracy for most classes. Most methods have 0% accuracy for class 9 since there are too few training samples in this class. The overall



(a) SVM, OA = 64.94% (b) SRC, OA = 71.17% (c) SRC-JS, OA = 76.41% (d) SRC-LP, OA = 79.40% (e) ODL, OA = 71.04%

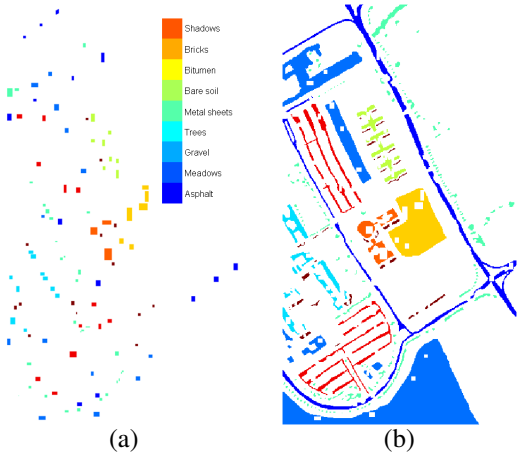
(f) ODL-JS, OA = 88.36% (g) ODL-LP, OA = 91.39% (h) TDDL, OA = 81.43% (i) TDDL-JS, OA = 92.65% (j) TDDL-LP, OA = 94.20%

Fig. 4: Classification map of the Indian Pine image obtained by (a) SVM, (b) SRC, (c) SRC-JS, (d) SRC-LP, (e) ODL, (f) ODL-JS, (g) ODL-LP, (h) TDDL, (i) TDDL-JS and (j) TDDL-LP.

TABLE III: Classification accuracy (%) for the Indian Pine image

Dictionary Size		$N = 997$			$N = 80$					
Class	SVM	SRC	SRC-JS	SRC-LP	ODL	ODL-JS	ODL-LP	TDDL	TDDL-JS	TDDL-LP
1	77.08	68.75	79.17	82.42	75.00	**97.92**	70.83	50.00	35.42	56.25
2	84.96	58.84	81.94	81.34	59.69	91.24	94.26	84.03	**94.57**	93.95
3	62.67	24.40	56.67	47.35	62.93	81.20	84.40	69.73	84.13	**92.13**
4	8.57	49.52	27.62	49.76	23.81	47.62	61.90	14.76	**79.05**	46.19
5	77.18	81.88	85.46	83.96	82.55	**93.29**	92.62	89.04	90.16	90.83
6	91.82	96.88	98.36	97.48	88.24	99.55	98.96	98.66	**99.55**	98.96
7	13.04	0.00	0.00	0.00	4.35	**17.39**	0.00	0.00	0.00	**95.65**
8	96.59	96.59	**100.00**	99.55	96.36	99.32	99.32	99.09	100.00	100.00
9	0.00	**5.56**	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
10	71.30	24.00	18.94	31.89	67.51	77.73	91.04	72.90	90.13	**94.03**
11	35.25	96.22	91.63	94.58	67.94	88.25	94.10	85.46	96.22	**97.37**
12	42.39	32.97	45.29	64.68	80.62	88.59	83.15	59.06	86.78	**95.47**
13	91.05	98.95	99.47	99.48	95.79	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
14	94.85	98.97	98.97	99.49	87.20	97.77	99.14	98.11	**99.40**	**99.40**
15	30.70	49.71	55.85	63.84	32.16	70.76	67.84	47.66	77.78	**82.75**
16	27.06	88.24	95.29	97.65	69.41	96.47	85.88	92.94	91.76	**98.82**
OA[%]	64.94	71.17	76.41	79.40	71.04	88.36	91.39	81.43	92.65	**94.20**
AA[%]	56.53	60.72	64.67	64.67	62.10	77.94	82.18	66.43	76.56	**83.86**
κ	0.647	0.695	0.737	0.712	0.691	0.851	0.907	0.8087	0.924	**0.940**

performance of TDDL-JS and TDDL-LP have at least 13% improvement over the other conventional dictionary learning techniques. TDDL-LP significantly outperforms other methods on the classes that occupy small regions in the image. The class 7 (Grass/Pasture-mowed), lying in a non-homogeneous region, has only 3 training samples and 23 test samples. The TDDL-LP is capable of correctly classify 95.65% test samples while the second highest accuracy is only 17.39%. We notice that the AA of both ODL-LP (82.18%) and TDDL-LP (83.86%) are at least 4% higher than that of the other methods. This also suggests that the Laplacian-sparsity-enforced dictionary learning methods work better on non-homogeneous regions, since the AA can only attain high value when both the most regions reach high accuracy.



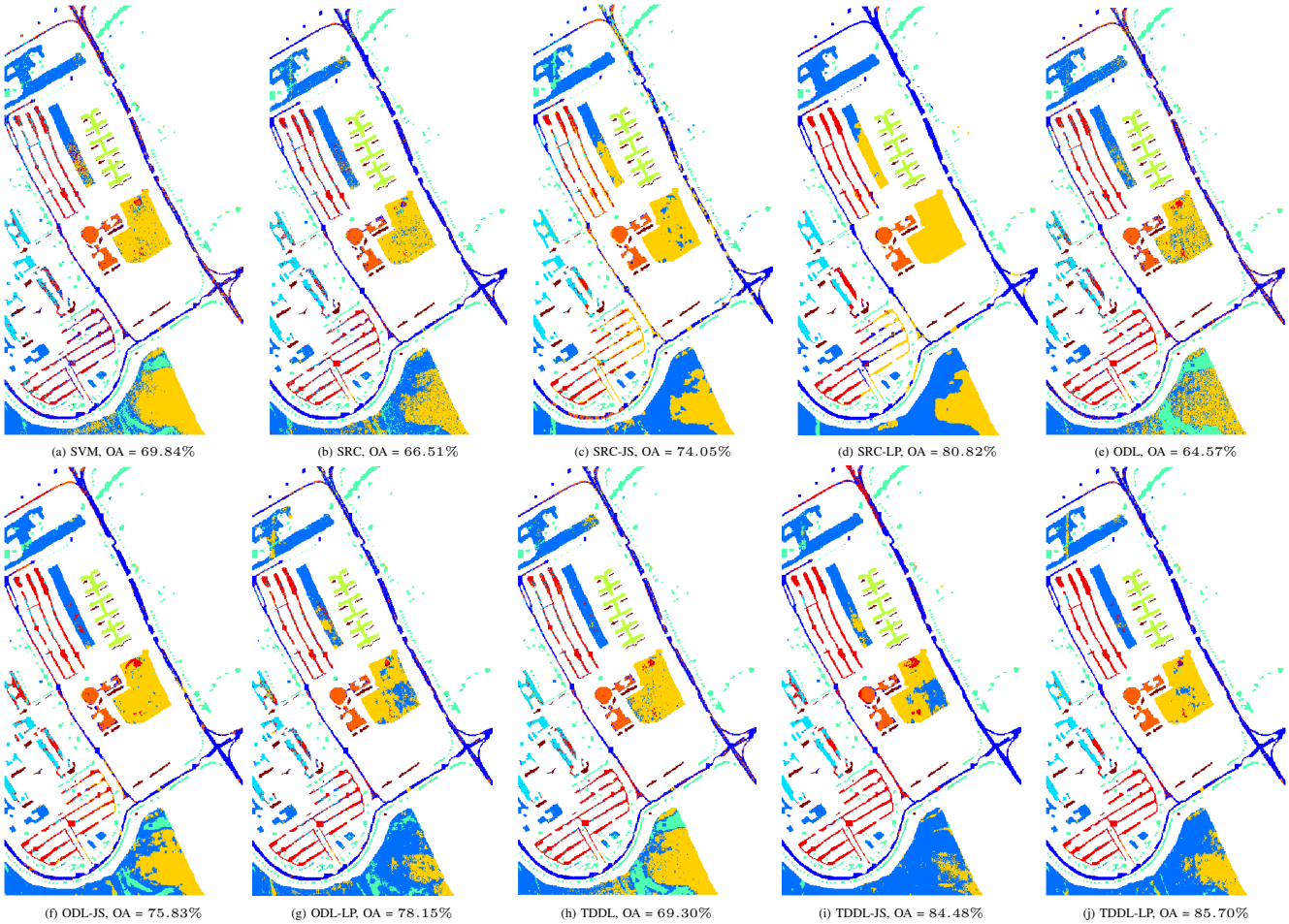
(a) (b)

Fig. 5: (a) Training sets and (b) test sets of the University of Pavia image.

C. Classification on ROSIS Pavia Urban Data Set

The last two images to be tested are the University of Pavia and the Center of Pavia, which are urban images acquired by the Reflective Optics System Imaging Spectrometer (ROSIS). It generates 115 spectral bands ranging from 0.43 to 0.86μm.

The University of Pavia image contains 610 $\times$ 340 pixels. 12 noisiest bands out of all 115 bands are removed. There are nine ground-truth classes of interests as shown in Table IV. For this image, the training samples were manually labelled



(a) SVM, OA = 69.84% (b) SRC, OA = 66.51% (c) SRC-JS, OA = 74.05% (d) SRC-LP, OA = 80.82% (e) ODL, OA = 64.57%

(f) ODL-JS, OA = 75.83% (g) ODL-LP, OA = 78.15% (h) TDDL, OA = 69.30% (i) TDDL-JS, OA = 84.48% (j) TDDL-LP, OA = 85.70%

Fig. 6: Classification map of the University of Pavia image obtained by (a) SVM, (b) SRC, (c) SRC-JS, (d) SRC-LP, (e) ODL, (f) ODL-JS, (g) ODL-LP, (h) TDDL, (i) TDDL-JS and (j) TDDL-LP.

TABLE V: Classification accuracy (%) for the University of Pavia image

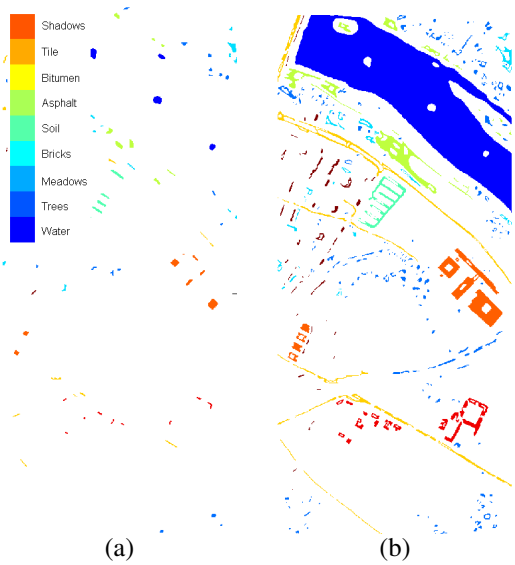
Dictionary Size		$N = 3921$			$N = 45$					
Class	SVM	SRC	SRC-JS	SRC-LP	ODL	ODL-JS	ODL-LP	TDDL	TDDL-JS	TDDL-LP
1	84.55	57.11	77.04	**95.08**	39.16	86.64	79.38	74.60	79.27	87.77
2	82.45	58.22	67.98	66.70	66.37	56.48	75.89	51.27	**86.85**	78.89
3	77.08	57.33	44.32	77.55	65.40	**80.72**	62.42	77.19	71.13	78.79
4	94.19	95.94	95.13	95.19	78.67	**99.04**	96.91	98.08	98.87	98.21
5	99.01	**100.00**	99.85	**100.00**	99.91	100.00	99.82	99.91	99.91	99.91
6	23.55	89.60	88.31	96.60	64.94	**96.89**	72.13	90.07	68.74	91.64
7	2.06	83.27	96.59	**96.59**	91.64	91.23	84.10	86.14	68.09	93.17
8	33.89	48.65	65.20	67.36	67.36	90.81	75.98	78.00	95.54	**94.20**
9	53.05	93.69	99.59	**99.59**	71.07	98.37	93.46	95.72	91.82	95.09
OA[%]	69.84	66.51	74.05	80.82	64.57	75.83	78.15	69.30	84.48	**85.70**
AA[%]	61.09	75.98	80.06	88.80	71.66	88.91	82.23	83.44	84.47	**90.85**
κ	0.569	0.628	0.681	0.758	0.549	0.731	0.747	0.662	0.817	**0.835**

by an analyst. The total number of training and testing samples is $3,921$ (10.64% of all the interested pixels) and $40,002$ respectively. The training and testing map are visually displayed in Fig. 5.

For the University of Pavia, we set the total iterations of unsupervised and supervised dictionary learning methods to be 30 and 200 respectively. The window size is set to 5×5 for all joint or Laplacian sparse regularized methods to obtain the highest OA. The ODL-LP is able to reach a performance of 78.15% for OA, which is more than 14% higher than that of ODL. The ODL-JS also significantly improves the OA, which is more than 11% higher than that of ODL. TDDL-LP has the highest OA = 85.70%, which indicates that it outperforms other methods when classify large regions of the image. It also has the highest $\kappa = 0.935$. The best classification accuracy for class 1 (Asphalt), which consists of narrow strips, is obtained by using TDDL-LP (87.77%). Class 2 (Meadows) is composed of large smooth regions, as expected, TDDL-JS gives the highest accuracy (86.85%) for this class. TDDL has large amount of misclassification pixels for class 2. The highest AA (90.85%) is given by TDDL-LP, which confirms that the TDDL-LP is superior to other methods when classify

TABLE IV: Number of training and test samples for the University of Pavia image

Class #	Name	Train	Test
1	Asphalt	548	6304
2	Meadows	540	18146
3	Gravel	392	1815
4	Trees	524	2912
5	Metal sheets	265	1113
6	Bare soil	532	4572
7	Bitumen	375	981
8	Bricks	514	3364
9	Shadows	231	795
Total		3921	40002



(a) (b)

Fig. 7: (a) Training sets and (b) test sets of the Center of Pavia image.

TABLE VI: Number of training and test samples for the Center of Pavia image

Class #	Name	Train	Test
1	Water	745	64533
2	Trees	785	5722
3	Meadows	797	2094
4	Bricks	485	1667
5	Soil	820	5729
6	Asphalt	678	6847
7	Bitumen	808	6479
8	Tile	223	2899
9	Shadows	195	1970
Total		5536	97940

the pixels in non-homogeneous regions.

The third image where we evaluate various approaches is the Center of Pavia, which consists of 1094×492 pixels. Each pixel has 102 bands after removing 13 noisy bands. This image consists of nine ground-truth classes of interest as shown in Table VI and Fig. 7. $5,536$ manually labelled pixels are designated as the training samples and the remaining $97,940$ interested pixels are used for testing.

Since this image has more labeled samples than the other two images, we set the total iterations of unsupervised and supervised dictionary learning methods to be 75 and 1000 respectively. The window size is set to 5×5 for the joint sparse and Laplacian sparse regularized methods. Although the OA of most methods are close, the OA of ODL-JS and ODL-LP are still around 3% higher than that of ODL. The TDDL-LP reach the highest OA = 98.67% over all the other methods. The OA of TDDL-JS (98.01%) is slightly lower than that of the TDDL-LP. We notice that SRC-JS (OA = 98.01%) and SRC-LP (OA=98.36%) also render competitive performance when compared to TDDL-JS and TDDL-LP due to the fact that the raw spectral features of this image is already highly discriminative. TDDL-LP outperforms other methods on almost all classes and works especially well for Class 4 (Bricks), achieving highest accuracy of 97.41%. Except for SRC-LP where the accuracy is 94.72%, none of others reaches accuracy over 90% for Class 4. Additionally, the AA of TDDL-LP (97.21%) is almost 2% better than that of TDDL-JS (95.68%). These results support our assertion that the Laplacian sparsity prior provides stronger discriminability on nonhomogeneous regions. Performance comparison between the SRC-based and TDDL-based methods have shown that the dictionary size can be drastically decreased by applying supervised dictionary learning while achieving even better performance.

VI. CONCLUSION

In this paper, we proposed novel a task driven dictionary learning method with joint or Laplacian sparsity prior for HSI classification. The corresponding optimization algorithms are developed using fixed point differentiation, and are further simplified for ease of implementation. We also derived the optimization algorithm for solving the Laplacian sparse recovery problem using SpaRSA, which improves the computational efficiency due to the availability of a more accurate descent direction. The performance and the behavior of the proposed methods, i.e. TDDL-JS and TDDL-LP, have been extensively studied on the popular hyperspectral images. The results confirm that both TDDL-JS and TDDL-LP give plausible results on smooth homogeneous regions, while TDDL-LP one works better for classifying small narrow regions. Compared to TDDL-JS, TDDL-LP is able to obtain a more stable performance by describing the similarities of neighboring pixels' sparse codes more delicately. The results also confirm that a significantly better performance can still be achieved when joint or Laplacian prior is imposed by using a very small dictionary. The overall accuracy of our algorithm can be improved by applying kernelization to the proposed approach. This can be achieved by kernelizing the sparse representation [44] and using a composite kernel classifier [45].

VII. APPENDIX A

We can infer from Eq. (18) that $vec\left(\partial \mathbf{A}/\partial D_{mn}\right) = \mathbf{0}, \forall n \in \Lambda^c$, which indicates $\partial \mathcal{L}/\partial D_{mn} = \mathbf{0}$, $\forall n \in \Lambda^c$. Therefore, we only need to take the gradient $\partial \mathcal{L}/\partial D_{mn}$, $\forall n \in \Lambda$ into account.

From the Eq. (13) and Eq. (18), we achieve the gradient for every element of $\tilde{\mathbf{D}}$,

$$\frac{\partial \mathcal{L}}{\partial \tilde{D}_{mn}} = vec\left(\frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{A}}}\right)^{\top} \cdot vec\left(\frac{\partial \tilde{\mathbf{A}}}{\partial \tilde{D}_{mn}}\right), \qquad (32)$$

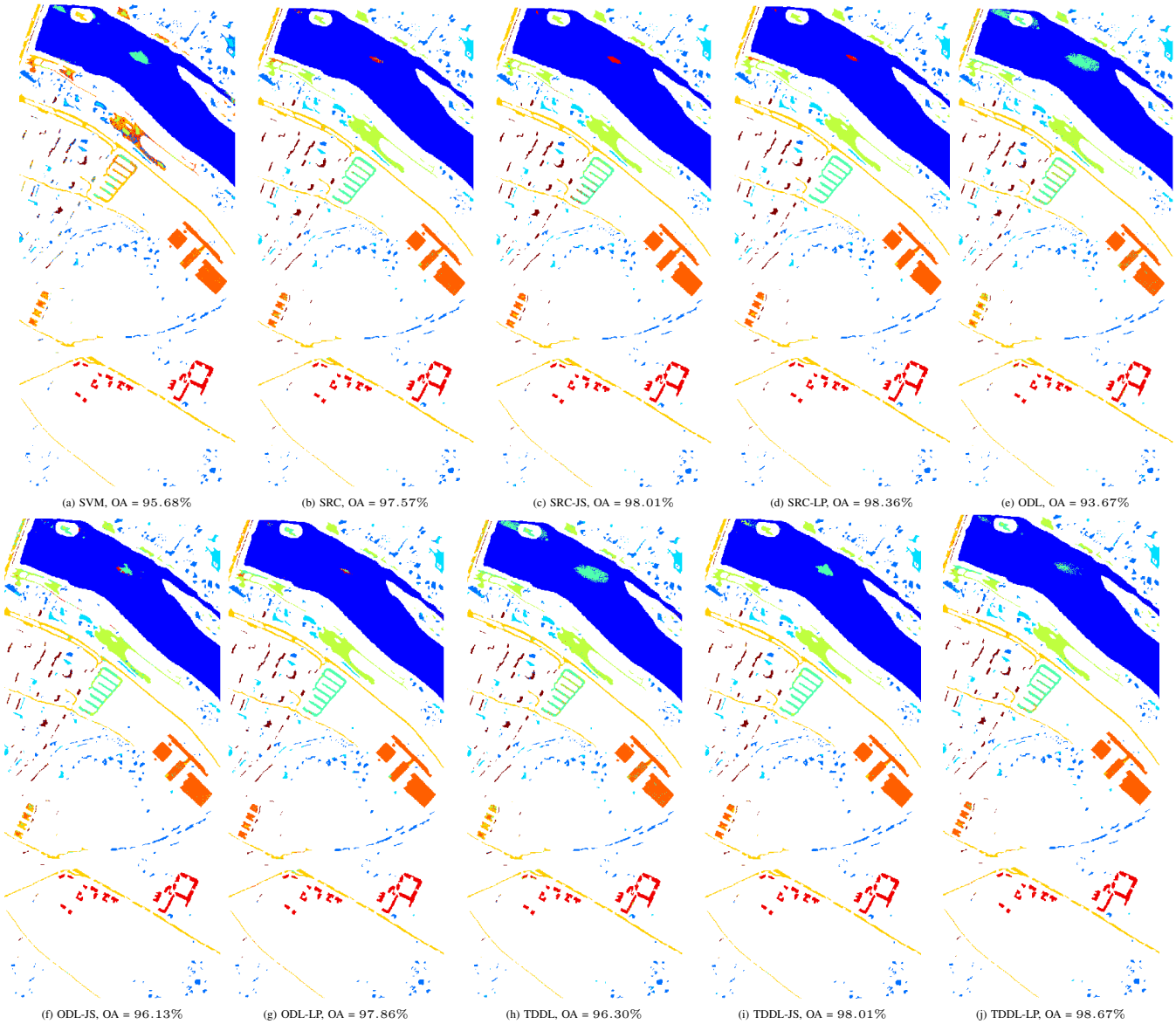


Fig. 8: Classification map of the Center of Pavia image obtained by (a) SVM, (b) SRC, (c) SRC-JS, (d) SRC-LP, (e) ODL, (f) ODL-JS, (g) ODL-LP, (h) TDDL, (i) TDDL-JS and (j) TDDL-LP.

where $m = 1, \ldots, M$ and $n = 1, \ldots, N_\Lambda$. Let $\mathbf{g} = vec\left(\frac{\partial f}{\partial \tilde{\mathbf{A}}^\top}\right) = vec\left(\left(\mathbf{W}\hat{\mathbf{A}} - \hat{\mathbf{Y}}\right)^\top \tilde{\mathbf{W}}\right)$ and $\tilde{\mathbf{W}} = \mathbf{W}^\Lambda$ is the Λ columns of $\mathbf{W}$. Expand Eq. (18) and combine it with Eq. (32), we have

$$\frac{\partial \mathcal{L}}{\partial \tilde{D}_{mn}} = U_{mn} - V_{mn} \text{ and } \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{D}}} = \mathbf{U} - \mathbf{V}, \tag{33}$$

where $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{m \times N_\Lambda}$ and every element U_{mn}, V_{mn} are defined as

$$U_{mn} = \mathbf{g}^\top \left(\tilde{\mathbf{D}}^\top \tilde{\mathbf{D}} \otimes \mathbf{I}_P + \lambda \mathbf{\Gamma}\right)^{-1} vec\left((\mathbf{X} - \mathbf{D}\mathbf{A})^\top \tilde{\mathbf{E}}_{mn}\right),$$
$$V_{mn} = \mathbf{g}^\top \left(\tilde{\mathbf{D}}^\top \tilde{\mathbf{D}} \otimes \mathbf{I}_P + \lambda \mathbf{\Gamma}\right)^{-1} vec\left(\tilde{\mathbf{A}}^\top \tilde{\mathbf{E}}_{mn}^\top \tilde{\mathbf{D}}\right),$$

where $\tilde{\mathbf{E}}_{mn} \in \mathbb{R}^{M \times N_\Lambda}$ is the indicator matrix that element (m, n) of $\tilde{\mathbf{E}}_{mn}$ is 1 and all other elements are zero.

Consider the simplification for $\mathbf{U}$ first

$$\begin{aligned} U_{mn} &= \mathbf{g}^\top \left(\tilde{\mathbf{D}}^\top \tilde{\mathbf{D}} \otimes \mathbf{I}_P + \lambda \mathbf{\Gamma}\right)^{-1} \left(\tilde{\mathbf{E}}_{mn}^\top \otimes \mathbf{I}_P\right) vec\left((\mathbf{X} - \mathbf{D}\mathbf{A})^\top\right) \\ &= \mathbf{g}^\top \mathbf{F}^{\tilde{n}} vec\left((\mathbf{X} - \mathbf{D}\mathbf{A})^\top\right)_{\tilde{m}}, \end{aligned} \tag{34}$$

where $\mathbf{F} = \left(\tilde{\mathbf{D}}^\top \tilde{\mathbf{D}} \otimes \mathbf{I}_P + \lambda \mathbf{\Gamma}\right)^{-1}$; $\tilde{m}(m) = \{(m-1)P + 1, \ldots, mP\}$, $\tilde{n}(n) = \{(n-1)P+1, \ldots, nP\}$ denote the index sets; $\mathbf{F}^{\tilde{n}}$ are the $\tilde{n}$ columns of $\mathbf{F}$.

Let $\boldsymbol{\xi}_m = vec\left((\mathbf{X} - \mathbf{D}\mathbf{A})^\top\right)_{\tilde{m}}$. It can be shown that $\boldsymbol{\xi}_m^\top$ is the m^{th} row of $(\mathbf{X} - \mathbf{D}\mathbf{A})$. Now the (m, n) element U_{mn} of the first part $\mathbf{U}$ can be written as

$$U_{mn} = \left(\mathbf{g}^\top \mathbf{F}\right)^{\tilde{n}} \boldsymbol{\xi}_m, \tag{35}$$

TABLE VII: Classification accuracy (%) for the Center of Pavia image

Dictionary Size		$N = 5536$			$N = 45$					
Class	SVM	SRC	SRC-JS	SRC-LP	ODL	ODL-JS	ODL-LP	TDDL	TDDL-JS	TDDL-LP
1	96.97	99.58	99.52	99.28	96.26	99.13	**99.69**	98.54	98.76	99.13
2	91.09	90.07	96.89	92.11	84.25	94.63	90.63	89.55	**97.59**	93.01
3	96.08	95.42	99.47	98.62	93.36	96.23	97.61	95.18	96.85	**98.71**
4	86.32	79.96	78.28	94.72	61.61	64.73	97.30	85.78	85.18	**97.41**
5	88.57	93.70	97.05	97.14	89.40	84.62	90.00	88.08	98.25	**99.59**
6	95.27	95.62	98.19	97.18	94.35	95.03	94.49	94.39	**99.36**	99.18
7	94.03	93.86	97.01	96.84	86.31	86.90	97.33	91.65	94.46	**98.66**
8	99.83	99.17	99.66	99.66	96.76	99.79	99.00	98.17	99.38	**99.73**
9	85.74	98.58	99.19	**99.95**	93.25	90.56	94.42	95.53	91.27	95.61
OA[%]	95.68	97.57	98.01	98.36	93.67	96.13	97.86	96.30	98.01	**98.67**
AA[%]	93.77	94.00	95.03	97.28	88.39	90.18	95.61	92.99	95.68	**97.89**
κ	0.923	0.961	0.965	0.971	0.899	0.938	0.965	0.940	0.968	**0.979**

Stacking all elements of $\mathbf{U}$

$$\begin{aligned}\mathbf{U} &= \begin{bmatrix} \left(\mathbf{g}^\top\mathbf{F}\right)^{\tilde{1}}\boldsymbol{\xi}_1 & \cdots & \left(\mathbf{g}^\top\mathbf{F}\right)^{\widetilde{N_\Lambda}}\boldsymbol{\xi}_1 \\ \vdots & \ddots & \vdots \\ \left(\mathbf{g}^\top\mathbf{F}\right)^{\tilde{1}}\boldsymbol{\xi}_M & \cdots & \left(\mathbf{g}^\top\mathbf{F}\right)^{\widetilde{N_\Lambda}}\boldsymbol{\xi}_M \end{bmatrix} \\ &= \boldsymbol{\xi}\left[(\mathbf{g}^\top\mathbf{F})^{\tilde{1}^\top}\cdots(\mathbf{g}^\top\mathbf{F})^{\tilde{N_\Lambda}^\top}\right], \end{aligned} \tag{36}$$

where Λ_n denotes the n^{th} element of set Λ.

Now consider simplification for $\mathbf{V}$. Each element V_{mn} of $\mathbf{V}$ can be written as

$$\begin{aligned} V_{mn} &= \mathbf{g}^\top\mathbf{F}\cdot vec\left(\tilde{\mathbf{A}}^\top\tilde{\mathbf{E}}_{mn}^\top\tilde{\mathbf{D}}\right) \\ &= \mathbf{g}^\top\mathbf{F}\left(\tilde{\mathbf{D}}^\top\tilde{\mathbf{E}}_{mn}\otimes\mathbf{I}_P\right)vec\left(\tilde{\mathbf{A}}^\top\right) \\ &= \mathbf{g}^\top\mathbf{F}\left(\tilde{\mathbf{D}}_m^\top\otimes\mathbf{I}_P\right)\tilde{\mathbf{A}}_n^\top, \end{aligned} \tag{37}$$

where $\tilde{\mathbf{A}}_n$ is the n^{th} row of $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{D}}_m$ is the m^{th} row of $\mathbf{D}$. Stacking every element of $\mathbf{V}$, such that

$$\begin{aligned}\mathbf{V} &= \begin{bmatrix} \mathbf{g}^\top\mathbf{F}\left(\tilde{\mathbf{D}}_1^\top\otimes\mathbf{I}_P\right) & \cdots & \mathbf{g}^\top\mathbf{F}\left(\tilde{\mathbf{D}}_1^\top\otimes\mathbf{I}_P\right) \\ \vdots & \ddots & \vdots \\ \mathbf{g}^\top\mathbf{F}\left(\tilde{\mathbf{D}}_M^\top\otimes\mathbf{I}_P\right) & \cdots & \mathbf{g}^\top\mathbf{F}\left(\tilde{\mathbf{D}}_M^\top\otimes\mathbf{I}_P\right) \end{bmatrix}\mathbf{A}^\top \\ &= \begin{bmatrix} \sum_{n=1}^N \mathbf{D}_{1n}^\top\left(\mathbf{g}^\top\mathbf{F}\right)_{\bar{n}}\mathbf{A}_1^\top & \cdots & \sum_{n=1}^N \mathbf{D}_{1n}^\top\left(\mathbf{g}^\top\mathbf{F}\right)_{\bar{n}}\mathbf{A}_N^\top \\ \vdots & \ddots & \vdots \\ \sum_{n=1}^N \mathbf{D}_{Mn}^\top\left(\mathbf{g}^\top\mathbf{F}\right)_{\bar{n}}\mathbf{A}_1^\top & \cdots & \sum_{n=1}^N \mathbf{D}_{Mn}^\top\left(\mathbf{g}^\top\mathbf{F}\right)_{\bar{n}}\mathbf{A}_N^\top \end{bmatrix} \\ &= \mathbf{D}\left[\left(\mathbf{g}^\top\mathbf{F}\right)_{\bar{1}}^\top\ldots\left(\mathbf{g}^\top\mathbf{F}\right)_{\bar{P}}^\top\right]\mathbf{A}^\top, \end{aligned} \tag{38}$$

where $\bar{p}(p) = \{p, p+P, \ldots, p+(N-1)P\}$. Combining Eq. (36) and Eq. (38)

$$\frac{\partial\mathcal{L}}{\partial\tilde{\mathbf{D}}} = \mathbf{U}-\mathbf{V} = \boldsymbol{\xi}\boldsymbol{\beta}_\Lambda^\top - \tilde{\mathbf{D}}\boldsymbol{\beta}_\Lambda\tilde{\mathbf{A}}^\top \text{ and } \frac{\partial\mathcal{L}}{\partial\mathbf{D}} = \boldsymbol{\xi}\boldsymbol{\beta}^\top - \mathbf{D}\boldsymbol{\beta}\mathbf{A}^\top, \tag{39}$$

where $\boldsymbol{\beta}_{\Lambda_c} = \mathbf{0}$ and $\boldsymbol{\beta}_\Lambda = [\left(\mathbf{g}^\top\mathbf{F}\right)_{\tilde{1}}^\top, \cdots, \left(\mathbf{g}^\top\mathbf{F}\right)_{\widetilde{N_\Lambda}}^\top]^\top$. More generally, we have defined $\boldsymbol{\beta}_\Lambda \in \mathbb{R}^{N_\Lambda\times P}$ such that $vec(\boldsymbol{\beta}_\Lambda^\top) = \mathbf{F}\mathbf{g}$.

VIII. Appendix B

The gradient for updating the dictionary can be written as

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial D_{mn}} &= vec\left(\frac{\partial\mathcal{L}}{\partial\mathbf{A}}\right)^\top\cdot vec\left(\frac{\partial\mathbf{A}}{\partial D_{mn}}\right) \\ &= vec\left(\frac{\partial\mathcal{L}}{\partial\mathbf{A}}\right)_\Lambda^\top\cdot vec\left(\frac{\partial\mathbf{A}}{\partial D_{mn}}\right)_\Lambda, \end{aligned} \tag{40}$$

Expand Eq. (31) and combine it with Eq. (40), the desired gradient is

$$\frac{\partial\mathcal{L}}{\partial D_{mn}} = U_{mn} - V_{mn} \text{ and } \frac{\partial\mathcal{L}}{\partial\mathbf{D}} = \mathbf{U}-\mathbf{V}, \tag{41}$$

where

$$\begin{aligned} U_{mn} &= \mathbf{g}^\top\mathbf{F}^{-1}vec\left(\mathbf{E}_{mn}^\top\left(\mathbf{X}-\mathbf{D}\mathbf{A}\right)\right)_\Lambda, \\ V_{mn} &= \mathbf{g}^\top\mathbf{F}^{-1}vec\left(\mathbf{D}^\top\mathbf{E}_{mn}\mathbf{A}\right)_\Lambda, \\ \mathbf{F} &= \left(\mathbf{I}_P\otimes\mathbf{D}^\top\mathbf{D}+\gamma\mathbf{L}\otimes\mathbf{I}_N\right)_{\Lambda,\Lambda}^{-1}. \end{aligned}$$

Let $\mathbf{g}$ has the same definition as that in Section VII. The first part $\mathbf{U}$ of $\frac{\partial f}{\partial D_{mn}}$ is

$$\begin{aligned} U_{mn} &= \mathbf{g}^\top\mathbf{F}vec\left(\mathbf{E}_{mn}^\top\left(\mathbf{X}-\mathbf{D}\mathbf{A}\right)\right)_\Lambda \\ &= \left(\mathbf{g}^\top\mathbf{F}\right)_{\tilde{n}}vec\left(\mathbf{X}-\mathbf{D}\mathbf{A}\right)_{\tilde{m}(m,n)} \end{aligned} \tag{42}$$

$\mathbf{E}_{mn}\in\mathbb{R}^{M\times N}$ is the indicator matrix that the (m,n) element is 1 and all other elements are zero. $\tilde{m}$ and $\tilde{n}$ are defined as the following index sets,

$$\begin{aligned}\tilde{m}(m,n) &= \{m,\ldots,m+pM,\ldots\},\forall p \text{ s.t. } n+pN\in\Lambda \\ \tilde{n}(n) &= \{n,n+N,\ldots,n+(P-1)N\}\cap\Lambda\end{aligned}$$

Eq. (42) can be further simplified by introducing $\mathbf{h}^{(n)}\in\mathbb{R}^P$, such that

$$\mathbf{h}^{(n)} = \begin{cases}\left(\mathbf{g}^\top\mathbf{F}\right)_{n+pN}, \text{ if } n+pN\in\tilde{n}(n),\forall p \\ 0, \text{ otherwise}\end{cases} \tag{43}$$

Now Eq. (42) can be rewritten as,

$$U_{mn} = \mathbf{h}^{(n)\top}\boldsymbol{\xi}_m, \tag{44}$$

where $\boldsymbol{\xi}_m^\top$ is the m^{th} row of $\mathbf{X}-\mathbf{D}\mathbf{A}$. The first part $\mathbf{U}$ of the gradient $\frac{\partial f}{\partial\mathbf{D}}$ can be obtained by stacking all U_{mn} in Eq. (44)

$$\begin{aligned}\mathbf{U} &= \begin{bmatrix}\boldsymbol{\xi}_1^\top\mathbf{h}^{(1)} & \cdots & \mathbf{x}_1^\top\mathbf{h}^{(N)} \\ \vdots & \ddots & \vdots \\ \boldsymbol{\xi}_M^\top\mathbf{h}^{(1)} & \cdots & \boldsymbol{\xi}_M^\top\mathbf{h}^{(N)}\end{bmatrix} \\ &= \boldsymbol{\xi}\left[\mathbf{h}^{(1)}\cdots\mathbf{h}^{(N)}\right] \\ &= \boldsymbol{\xi}\boldsymbol{\beta}^\top, \end{aligned} \tag{45}$$

where we define $\boldsymbol{\beta} = \left[\mathbf{h}^{(1)}, \cdots, \mathbf{h}^{(N)}\right]^\top \in \mathbb{R}^{N\times P}$. By examining the nonzero elements position of $\mathbf{h}^{(1)}, \ldots, \mathbf{h}^{(N)}$, it is not difficult to find the relation between $\boldsymbol{\beta}$ and $\mathbf{g}^\top\mathbf{F}$

$$vec\left(\boldsymbol{\beta}\right)_\Lambda = \mathbf{Fg} \text{ and } vec\left(\boldsymbol{\beta}\right)_{\Lambda^c} = \mathbf{0}. \tag{46}$$

Now consider the second term V_{mn} of $\frac{\partial f}{\partial D_{mn}}$

$$\begin{aligned} V_{mn} &= \left(\mathbf{g}^\top\mathbf{F}\right)\left(\mathbf{I}_P \otimes \mathbf{D}^\top\mathbf{E}_{mn}\right)_{\Lambda,\Lambda} vec\left(\mathbf{A}\right)_\Lambda \\ &= \left(\mathbf{g}^\top\mathbf{F}\right)\left(\left[\mathbf{D}_m vec(\mathbf{A})_n \ldots \mathbf{D}_m vec(\mathbf{A})_{(n+(P-1)N)}\right]^\top\right)_\Lambda \\ &= \mathbf{D}_m \sum_{p=1}^{P} \mathbf{A}_{n,p}\boldsymbol{\beta}^p, \end{aligned} \tag{47}$$

where $\boldsymbol{\beta}^p$ is the p^{th} column of $\boldsymbol{\beta}$. The differentiation $\frac{\partial f}{\partial \mathbf{D}}$ can be derived from V_{mn} in Eq. (47)

$$\begin{aligned} \mathbf{V} &= \begin{bmatrix} \mathbf{D}_1\left[\sum_{p=1}^P \mathbf{A}_{1,p}\boldsymbol{\beta}^p, \cdots, \sum_{p=1}^P \mathbf{A}_{N,p}\boldsymbol{\beta}^p\right] \\ \vdots \\ \mathbf{D}_M\left[\sum_{p=1}^P \mathbf{A}_{1,p}\left(\mathbf{Fg}\right)_{\hat{p}} \cdots \sum_{p=1}^P \mathbf{A}_{N,p}\left(\mathbf{Fg}\right)_{\hat{p}}\right] \end{bmatrix} \\ &= \mathbf{D}\left[\sum_{p=1}^{P} \mathbf{A}_{1,p}\boldsymbol{\beta}^p \cdots \sum_{p=1}^{P} \mathbf{A}_{N,p}\boldsymbol{\beta}^p\right] \\ &= \mathbf{D}\boldsymbol{\beta}\mathbf{A}^\top, \end{aligned} \tag{48}$$

Combining Eq. (45) and Eq. (48), we reach the gradient of the dictionary

$$\frac{\partial \mathcal{L}}{\partial \mathbf{D}} = \boldsymbol{\xi}\boldsymbol{\beta}^\top - \mathbf{D}\boldsymbol{\beta}\mathbf{A}^\top. \tag{49}$$

REFERENCES

[1] N. M. Nasrabadi, "Hyperspectral target detection," *IEEE Signal Process. Mag.*, vol. 31, no. 1, Jan. 2014.

[2] Y. Chen, N. M. Nasrabadi, and T. D. Tran, "Sparse representation for target detection in hyperspectral imagery," *IEEE Journal of Select. Topics in Signal Process.*, vol. 5, no. 3, pp. 629–640, Jun. 2011.

[3] D. Manolakis, "Detection algorithms for hyperspectral imaging applications: a signal processing perspective," *IEEE Workshop on Adv. in Tech. for Anal. of Remote. Sens. Data*, pp. 378–384, Oct. 2003.

[4] G. Camps-Valls, D. Tuia, L. Bruzzone, and J. Atli, "Advances in hyperspectral image classification: Earth monitoring with statistical learning methods," *IEEE Signal Process. Mag.*, vol. 31, no. 1, pp. 45–54, Jan. 2014.

[5] F. J. Herrmann, M. P. Friedlander, and O. Yilmaz, "Fighting the curse of dimensionality: compressive sensing in exploration seismology," *IEEE Signal Process. Mag.*, vol. 29, no. 3, pp. 88–100, May 2012.

[6] M. D. Iordache, J. M. Bioucas-Dias, and A. Plaza, "Sparse unmixing of hyperspectral data," *IEEE Trans. on Geosci. and Remote Sens.*, vol. 49, no. 6, pp. 2014–2039, Jun. 2011.

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

[8] L. Ma, M. M. Crawford, and J. Tian, "Local manifold learning-based k-nearest-neighbor for hyperspectral image classification," *IEEE Trans. on Geosci. and Remote Sens.*, vol. 48, no. 11, pp. 4099–4109, Nov. 2010.

[9] J. Li, J. Bioucas-Dias, and A. Plaza, "Semisupervised hyperspectral image segmentation using multinomial logistic regression with active learning," *IEEE Trans. on Geosci. and Remote Sens.*, vol. 48, no. 11, pp. 4085–4098, Nov. 2010.

[10] J. Benediktsson, P. Swain, and O. Ersoy, "Neural network approaches versus statistical methods in classification of multisource remote sensing data," *IEEE Trans. on Geosci. and Remote Sens.*, vol. 28, no. 4, pp. 540–552, Jul. 1990.

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

[12] X. Mei and H. Ling, "Robust visual tracking and vehicle classification via sparse representation," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 33, no. 11, pp. 2259–2272, Nov. 2011.

[13] J. Gemmeke, T. Virtanen, and A. Hurmalainen, "Exemplar-based sparse representations for noise robust automatic speech recognition," *IEEE Trans. on Audio, Speech, and Lan. Process.*, vol. 19, no. 7, pp. 2067–2080, Sept. 2011.

[14] A. Cheriyadat, "Unsupervised feature learning for aerial scene classification," *IEEE Trans. on Geosci. and Remote Sens.*, vol. 52, no. 1, pp. 439–451, Jan. 2014.

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

[16] J. Chen and X. Huo, "Theoretical results on sparse representations of multiple-measurement vectors," *Signal Processing, IEEE Transactions on*, vol. 54, no. 12, pp. 4634–4643, Dec. 2006.

[17] X. Sun, Q. Qu, N. M. Nasrabadi, and T. D. Tran, "Structured priors for sparse-representation-based hyperspectral image classification," *IEEE Geosci. and Remote Sens. Letters*, vol. 11, no. 7, pp. 1235–1239, Jul. 2014.

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

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

[20] S. Kim and E. P. Xing, "Tree-guided group lasso for multi-task regression with structured sparsity," *in ICML*, pp. 543–550, Jun. 2010.

[21] P. Sprechmann, I. Ramirez, G. Sapiro, and Y. Eldar, "C-Hilasso: A collaborative hierarchical sparse modeling framework," *IEEE Trans. on Signal Process.*, vol. 59, no. 9, pp. 4183–4198, Sept. 2011.

[22] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 35, no. 1, pp. 171–184, Jan. 2013.

[23] S. Gao, I. Tsang, and L. Chia, "Laplacian sparse coding, hypergraph laplacian sparse coding, and applications," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 35, no. 1, pp. 92–104, Jan. 2013.

[24] K. Engan, S. Aase, and J. Hakon Husoy, "Method of optimal directions for frame design," *in ICASSP*, vol. 5, pp. 2443–2446, Mar. 1999.

[25] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. on Signal Process.*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.

[26] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," *in ICML*, pp. 689–696, Jun. 2009.

[27] J. Mairal, F. Bach, and J. Ponce, "Task-driven dictionary learning," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 34, no. 4, pp. 791–804, Apr. 2012.

[28] J. Mairal, J. Ponce, G. Sapiro, A. Zisserman, and F. Bach, "Supervised dictionary learning," *in NIPS*, pp. 1033–1040, Dec. 2008.

[29] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Discriminative learned dictionaries for local image analysis," *in CVPR*, pp. 1–8, Jun. 2008.

[30] I. Ramirez, P. Sprechmann, and G. Sapiro, "Classification and clustering via dictionary learning with structured incoherence and shared features," *in CVPR*, pp. 3501–3508, Jun. 2010.

[31] J. Zhuolin, L. Zhe, and L. Davis, "Label consistent K-SVD: Learning a discriminative dictionary for recognition," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 35, no. 11, pp. 2651–2664, Nov. 2013.

[32] J. Yang, Z. Wang, Z. Lin, S. Cohen, and T. Huang, "Coupled dictionary training for image super-resolution," *IEEE Trans. on Image Process.*, vol. 21, no. 8, pp. 3467–3478, Aug. 2012.

[33] B. Colson, P. Marcotte, and G. Savard, "An overview of bilevel optimization," *Ann. of Operat. Res.*, vol. 153, no. 1, pp. 235–256, Apr. 2007.

[34] J. Yang, K. Yu, and T. Huang, "Supervised translation-invariant sparse coding," *in CVPR*, pp. 3517–3524, Jun. 2010.

[35] J. Zheng and Z. Jiang, "Tag taxonomy aware dictionary learning for region tagging," *in CVPR*, pp. 369–376, Jun. 2013.

[36] E. Alpaydin, *Introduction to Machine Learning*, 2nd ed. The MIT Press, 2010.

[37] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society, Series B*, vol. 67, pp. 301–320, Dec. 2005.

[38] D. M. Bradley and J. A. Bagnell, "Differentiable sparse coding," *in NIPS*, 2008.
[39] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2011.
[40] S. Wright, R. Nowak, and M. A. T. Figueiredo, "Sparse reconstruction by separable approximation," *IEEE Trans. on Signal Process.*, vol. 57, no. 7, pp. 2479–2493, Jul. 2009.
[41] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imag. Sciences*, vol. 2, no. 1, pp. 183–202, Jan. 2009.
[42] U. Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, Dec. 2007.
[43] J. Mairal, R. Jenatton, R. B. Francis, and R. O. Guillaume, "Network flow algorithms for structured sparsity," *in NIPS*, pp. 1558–1566, Dec. 2010.
[44] G. Camps-Valls, L. Gomez-Chova, J. Munoz-Mari, J. Vila-Frances, and J. Calpe-Maravilla, "Composite kernels for hyperspectral image classification," *IEEE Geosci. and Remote Sens. Letters*, vol. 3, no. 1, pp. 93–97, Jan 2006.
[45] Y. Chen, N. Nasrabadi, and T. Tran, "Hyperspectral image classification via kernel sparse representation," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 51, no. 1, pp. 217–231, Jan. 2013.