Log-Euclidean Kernel-Based Joint Sparse Representation for Hyperspectral Image Classification

Weidong Yang, Jiangtao Peng, Weiwei Sun, Member, IEEE, and Qian Du, Fellow, IEEE

Abstract—Motivated by the superior performance of region covariance descriptor, we use covariance matrices as new features to replace the original spectral pixel features, and employ a Log-Euclidean metric to characterize the geodesic distance between symmetric positive definite (SPD) covariance matrices. Based on the covariance features and Log-Euclidean metric, we propose a Log-Euclidean kernel-based joint sparse representation (LogEKJSR) model for the classification of hyperspectral images (HSIs). In the implementation of LogEKJSR, we first reduce the dimensionality of HSI by employing extended multiattribute profile (EMAP) transformations, and then extract the region covariance matrix features associated with each pixel on the EMAPs. The EMAP can model homogeneity and texture structure of HSI by aggregating multiple morphological attribute profiles, the covariance matrix feature contains both local spectral correlation and spatial structure information, and the Log-Euclidean kernel is a matrix-to-matrix similarity metric rather than vector-to-vector similarity metric. Finally, a LogEKJSR model is obtained by replacing the traditional kernel in the KJSR with the Log-Euclidean kernel. Experimental results on three benchmark hyperspectral data sets demonstrate that our proposed LogEKJSR is more effective than existing JSR, KJSR, and support vector machine methods.

Index Terms—Classification, covariance feature, hyperspectral image (HSI), joint sparse representation, Log-Euclidean kernel.

I. INTRODUCTION

Hyperspectral image (HSI) data contains a large number of spectral bands and can be represented as a cube, comprising one spectral dimension and two spatial dimensions. The spectral dimension provides reflections of materials at specific spectral bands. As the spectral characteristics of different materials are usually different, the pixelwise spectral features are discriminative and can be used to classify different materials. The spatial dimension provides spatial structure information around each pixel, which can be used to enhance the pixelwise spectral classification. Integrating the rich spectral and spatial information, various spectral-spatial methods have been proposed for the HSI classification [1]–[3], such as morphological profiles-based methods [4]–[6], composite kernels-based methods [7]–[12], deep learning methods [13], [14], and joint representation-based classifiers [15]–[21], etc.

Among the aforementioned methods, the joint sparse representation (JSR) method has attracted much attention due to its simplicity and effectiveness [15]–[19], [22]. The classical sparse representation classification (SRC) method represents a testing sample as a sparse linear combination of all training samples, and then classifies the testing sample to the class with the minimum reconstruction error. JSR improves the classical SRC method by simultaneously representing all spatial neighbors of a testing pixel using a common training dictionary and integrating the sparse representation of neighboring pixels for classification. JSR relies on the joint sparsity assumption that neighboring pixels are similar and have an equal contribution in the sparse representation. In order to improve the consistency of pixels in the neighborhood, many modified JSR methods are proposed either by weighting neighboring pixels, such as nonlocal-weighted JSR (WJSR) [16] and nearest-regularized JSR (NRJSR) [17], or by constructing adaptive neighborhood systems to replace the fixed squared neighborhood in the original JSR, such as image-segmentation-based adaptive local region [23], superpixel-based adaptive neighborhood [24], and shape-adaptive local smooth region [25].

JSR pursues a sparse linear representation for neighboring pixels and is essentially a linear model. Because hyperspectral data usually has nonlinear characteristics, the linear representation mechanism in the JSR model may be inaccurate. In order to cope with nonlinear problem, kernel-based JSR (KJSR) methods project the original HSI data into high dimensional feature space
and perform JSR on the projected data [26], [27]. By using the kernel trick [12], KJSR methods dramatically improve the original JSR [26]. However, kernel only measures the spectral similarity between different pixels and its discriminative ability may be insufficient for separating pixels with highly similar spectral characteristics. As HSI contains rich spatial information, incorporating spatial information into the kernel function is likely to further improve the discriminative performance of KJSR. An intuitive way to incorporate spatial information into the kernel function is to construct region features to replace the original spectral features and to measure region-to-region similarity, such as region kernel and superpixel-based kernel [9], [28], or set-to-set distance-based methods [29], [30]. Because HSI has local homogeneous distribution, pixels in a local region are similar. Therefore, region-to-region similarity is more accurate than pixel-to-pixel similarity [9]. Although these region-based methods have shown good performance, they usually focus on regions themselves but not region-based features.

Recently, region covariance descriptor has been proposed to characterize image patches [31], [32]. Rather than performing on regions themselves, region covariance descriptor computes the covariance matrix of a local region, which reflects the correlation of features in the region, i.e., its diagonal and nondiagonal elements corresponding to the variance of each feature and the correlations of different features, respectively. The covariance matrix features have proved to be effective for many applications, such as face recognition [33], object detection, and texture classification [31], [33]. The covariance representation has several advantages. First, it provides a natural way for fusing multiple features and produces a compact representation of original features [31]. Second, the size of covariance matrix is only related to the dimensionality of data. For regions with different sizes or shapes, the corresponding covariance matrices have the same size, which facilitates the comparison between any regions without being restricted to a constant window size [32].

Motivated by the superior performance of region covariance descriptor, we use covariance matrices as new features to replace the original spectral pixel features, and employ a Log-Euclidean metric to characterize the geodesic distance between symmetric positive definite (SPD) covariance matrices. Based on the covariance features and Log-Euclidean metric, we propose a Log-Euclidean kernel-based joint sparse representation (LogEKJSR) model for the classification of HSIs. In the implementation of LogEKJSR, we first reduce the dimensionality of HSI by employing extended multiattribute profile (EMAP) transformations, and then extract the region covariance matrix features associated with each pixel on the EMAPs. The EMAP can model homogeneity and texture structure of HSI by aggregating multiattribute elements corresponding to the variance of each feature and the correlations of different features, respectively. The covariance descriptor provides a compact representation of the original features and is proved to be highly discriminative for many image processing tasks, such as image classification, segmentation, and object recognition [33], [34].

If an image patch or region $R$ contains $n$ pixels, i.e., $p_1, p_2, \ldots, p_n$ with $p_i \in \mathbb{R}^d$, a $d \times d$ covariance matrix can be computed as

$$C_R = \frac{1}{n-1} \sum_{i=1}^{n} (p_i - \mu)(p_i - \mu)^T$$  \hspace{1cm} (1)

where $\mu$ is the mean of pixels in the region. Now, the local region $R$ is characterized by a single positive definite matrix $C_R$ (i.e., a covariance descriptor).

### B. Log-Euclidean Kernel

The covariance matrix $C_R$ is a $d \times d$ SPD matrix. Because the space of SPD matrices $S^+_d$ is not a linear space, the commonly used Euclidean distance metric can not be used [33], [34]. In fact, $S^+_d$ forms a Lie group that is a Riemannian manifold. In the Log-Euclidean framework, defining logarithmic multiplication and scalar logarithmic multiplication operators, $S^+_d$ is a complete inner product space [33]. The inner product between two SPD matrices $C_1$ and $C_2$ has the form

$$\langle C_1, C_2 \rangle_{\log} = \text{tr}(\log(C_1)\log(C_2))$$ \hspace{1cm} (2)

and the distance (metric) between them is

$$d_{\log}(C_1, C_2) = \| \log(C_1) - \log(C_2) \|_F$$ \hspace{1cm} (3)

Based on the inner product and Log-Euclidean distance, the linear, polynomial, exponential, and Gaussian Log-Euclidean
kernels are defined as [33]
\[
\kappa_i(C_1, C_2) = \langle C_1, C_2 \rangle_{\log} \\
\kappa_p(C_1, C_2) = p_n(\langle C_1, C_2 \rangle_{\log}) \\
\kappa_e(C_1, C_2) = \exp\left(p_n(\langle C_1, C_2 \rangle_{\log})\right) \\
\kappa_f(C_1, C_2) = \exp(-\beta d^2_{\log}(C_1, C_2))
\]
where \(p_n\) is an \(n\)-order polynomial, and \(\beta > 0\) is the Gaussian kernel parameter.

C. Kernel-Based Joint Sparse Representation (KJSR)

Given a testing pixel \(z\), we can collect all its neighbors in a \(w \times w\) spatial window centered at \(z\), and generate a neighborhood pixel matrix \(Z = [z_1, z_2, \ldots, z_T](T = w^2)\). Under the joint sparsity assumption, all neighboring pixels can be simultaneously represented by the training dictionary \(X = [x_1, x_2, \ldots, x_N]\):
\[
Z = [z_1, z_2, \ldots, z_T] = [X\alpha_1, X\alpha_2, \ldots, X\alpha_T] = XS
\]
where \(S = [\alpha_1, \alpha_2, \ldots, \alpha_T]\) is a sparse coefficient matrix and can be resolved by the following sparse recovery algorithm:
\[
\hat{S} = \arg \min_S \|Z - XS\|_F, \text{ s.t. } \|S\|_{\text{row},0} \leq K
\]
where \(\|S\|_{\text{row},0}\) refers to the number of nonzero rows of \(S\), and \(K\) is the sparsity level. The minimization problem (9) can be solved by the simultaneous orthogonal matching pursuit (SOMP) algorithm [15].

The JSR model in (8) can be kernelized by a feature map \(\phi\) as [26]
\[
Z_\phi = [\phi(z_1), \phi(z_2), \ldots, \phi(z_T)]
= [X_\phi\alpha_1, X_\phi\alpha_2, \ldots, X_\phi\alpha_T] = X_\phi S
\]
where \(X_\phi = [\phi(x_1), \phi(x_2), \ldots, \phi(x_N)] \in \mathcal{R}^{D \times N}\) and \(D\) is the dimensionality of the feature space. The row-sparse matrix \(S\) is recovered by solving the following problem:
\[
\hat{S} = \arg \min_S \|Z_\phi - X_\phi S\|_F^2, \text{ subject to } \|S\|_{\text{row},0} \leq K.
\]
The kernel SOMP (KSOMP) algorithm can approximately solve this kernel joint sparse recovery problem [26].

When the representation coefficient \(\hat{S}\) is obtained, the testing pixel \(z\) is classified to the class with the minimal approximation error.

III. LOG-EUCLIDEAN KERNEL-BASED JOINT SPARSE REPRESENTATION FOR HSI CLASSIFICATION

The framework of the proposed LogEKJSR algorithm is shown in Fig. 1. The EMAP features are first extracted from the original HSI. On the EMAP data, a local region for each pixel is constructed and the corresponding covariance matrix feature is computed. Based on the computed SPD covariance matrix features, the Log-Euclidean kernel can be computed and inputted to the kernel-based JSR model for the classification of HSI.

A. Covariance Feature Extraction for HSI

Rather than using the original pixels, we first extract the extended multiattribute profile (EMAP) features of HSI [4] and then compute the covariance matrix on the EMAPs.

In the construction of EMAPs, the principal component analysis (PCA) is first performed on the original HSI \(f\) to generate \(q\) principal component images \(g_1, g_2, \ldots, g_q\), and then attribute filtering operation with different attributes are performed on the principal component images to extract extended attribute profiles (EAPs), and finally all EAPs are merged to form EMAPs as
\[
\text{EMAP}(f) := \{\text{EAP}_1(f), \text{EAP}_2(f), \ldots, \text{EAP}_m(f)\}
\]
where \(\text{EAP}_i(f)\) is the EAP feature based on the attribute \(i\).
Here, the first three principal components are used (i.e., \( q = 3 \)), and three different attributes are considered (i.e., \( m = 3 \)) [6]: 1) \( a \), area of the regions; 2) \( i \), moment of inertia; and 3) \( s \), standard deviation of the gray-level values of the pixels in the regions. The threshold values for three attributes are adjusted based on the [5] as: \( \lambda_a = \{100, 500, 1000, 5000\} \), \( \lambda_i = \{0.2, 0.3, 0.4, 0.5\} \), and \( \lambda_s = \{20, 30, 40, 50\} \). It is easy to see that the dimensionality of EMAP is \( 3 \times 3 \times 9 = 81 \). In order to reduce the computational cost, we further reduce the dimensionality of EMAP to \( d = 20 \) by the PCA.

Then the compute covariance feature on EMAPs is computed to characterize the local neighborhood of a pixel. For each pixel \( x \in \mathbb{R}^d \) from the EMAP data, we find a \( wz \times wz \) local patch \( R_x \) centered at \( x \) and denote the pixels in the local region \( R_x \) as \( x_1, x_2, \ldots, x_T \), with \( T_1 = wz^2 \). Because pixels in the local region that build up the covariance descriptor may contain noise or have inconsistency characteristics, removing noise or outliers is beneficial. In order to keep the consistency of the region, we pick up \( M_1 \) pixels from the region that have smallest Euclidean distances to the central pixel \( x \) to form a new compact region \( \tilde{R}_x \). For example, for a pixel \( x \) in the Indian Pines HSI, we construct a \( 9 \times 9 \) spatial squared neighborhood centered at \( x \) as shown in Fig. 2, where the central pixel \( x \) is located in coordinate \( (5, 5) \) in the yellow region. In this neighborhood, there are 72 homogeneous pixels (the first eight rows in red region) and nine background pixels (the last row in blue region). If we set \( M_1 = 70 \), then 70 similar pixels (denoted as red circles) are selected to form a compact region \( \tilde{R}_x \) and 11 dissimilar pixels (denoted as green crosses) are deleted. It is clear that nine background pixels in the blue region are removed and the remaining pixels are homogeneous pixels.

The \( d \times d \) covariance matrix feature for the pixel \( x \) on the EMAP data can be computed as

\[
C_{\tilde{R}_x} = C_{\tilde{R}_x} + \epsilon \text{trace}(C_{\tilde{R}_x})I \tag{13}
\]

where \( \mu \) is the mean of pixels in the region \( \tilde{R}_x \). A simple regularization strategy is performed on the matrix \( C_{\tilde{R}_x} \) to avoid singularity as

\[
C_{\tilde{R}_x} = C_{\tilde{R}_x} + \epsilon \text{trace}(C_{\tilde{R}_x})I \tag{14}
\]

where \( I \) is an identity matrix, \( \epsilon \) is a small constant and set as \( \epsilon = 10^{-3} \) in the experiments. For the data in Fig. 2, we show the corresponding covariance matrix feature for all 81 pixels and the selected 70 pixels in Fig. 3. It can be clearly seen that the covariance feature in Fig. 3(b) is much more consistent than in Fig. 3(a) after inhomogeneous pixels being removed.

### B. Log-Euclidean Kernel-Based Joint Sparse Representation

In the JSR, it needs to utilize neighboring pixels of a testing pixel \( z \). Given a testing pixel \( z \), we find its neighbors in a \( wz_x \times wz_x \) window centered at \( z \). Here, we pick up \( M_2 \) most similar pixels (i.e., \( z_1, z_2, \ldots, z_{M_2} \)) to form neighborhood pixel set rather than using all \( T_2 = wz^2 \) neighboring pixels for the joint sparse representation of the testing pixel \( z \).

Let \( C_{z_k} (k = 1, \ldots, M_2) \) and \( C_{x_i} (i = 1, \ldots, N) \) be the covariance matrix features corresponding to the neighboring pixel \( z_k \) and training sample \( x_i \). Let \( \Phi \) be the function that maps SPD covariance matrices \( C_{z_k} \) or \( C_{x_i} \) to the reproducing kernel Hilbert space. The Log-EKJSR can be formulated as the following kernelized JSR problem:

\[
Z_{\Phi} = [\Phi(C_{z_1}), \Phi(C_{z_2}), \ldots, \Phi(C_{z_{M_2}})] = [X_{\Phi} \alpha_1, X_{\Phi} \alpha_2, \ldots, X_{\Phi} \alpha_{M_2}] = X_{\Phi} \hat{S} \tag{15}
\]

where \( X_{\Phi} = [\Phi(C_{x_1}), \Phi(C_{x_2}), \ldots, \Phi(C_{x_N})] \) is the feature representation of training set. The row-sparse matrix \( \hat{S} \) is recovered by solving the following problem:

\[
\hat{S} = \arg \min_{\bar{S}} \|Z_{\Phi} - X_{\Phi} \hat{S}\|_F, \text{subject to} \|\hat{S}\|_{row,0} \leq K. \tag{16}
\]

The KSOMP algorithm can approximately solve the problem (16) [26]. The classification of the testing pixel \( z \) is based on the reconstruction residuals as

\[
\text{Class}(z) = \arg \min_{r = 1, \ldots, C} r^c(z) \tag{17}
\]

where the reconstruction residual of the \( c \)th class is

\[
r^c(z) = \|Z_{\Phi} - (X_{\Phi} \cdot \Omega_{c}, \hat{S}_{\Omega_{c}})\|_F^2
\]

\[
= \sum_{k=1}^{M_2} \|\Phi(C_{z_k}) - (X_{\Phi} \cdot \Omega_{c}, \hat{S}_{\Omega_{c}, k})\|_F^2
\]

\[
= \sum_{k=1}^{M_2} \left( \kappa(C_{z_k}, C_{x_i}) - 2S_{\Omega_{c}, k}^T (K_{X, Z})_{\Omega_{c}, k} + \hat{S}_{\Omega_{c}, k}^T (K_{X, Z})_{\Omega_{c}, k} \right). \tag{18}
\]
Algorithm 1: LogEKJSR.

**Input:** Dictionary $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$, parameter $K$

**Output:** The label of all testing pixels.

1. Extract the EMAP features
2. Compute the covariance matrix features
3. Compute Log-Euclidean kernel $\mathbf{K}_{\mathbf{X}, \mathbf{X}}$
4. Perform KJSR for each testing pixel $\mathbf{z}$:
   1. Construct neighborhood matrix $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_{M_2}]$
   2. Compute Log-Euclidean kernels $\mathbf{K}_{\mathbf{Z}, \mathbf{Z}}$ and $\mathbf{K}_{\mathbf{X}, \mathbf{Z}}$
   3. Solve the coefficient matrix:
      \[
      \hat{S} = \arg\min_{\mathbf{S}} \| \mathbf{Z}_{\mathbf{f}} - \mathbf{X}_{\mathbf{f}} \mathbf{S} \|_{F}^2, \text{ s.t. } \| \mathbf{S} \|_{\text{row,0}} \leq K
      \]
   4. Compute the approximation error of each class:
      \[
      r^c(\mathbf{z}) = \| \mathbf{Z}_{\mathbf{f}} - (\mathbf{X}_{\mathbf{f}})_{\mathbf{c}} \hat{S}_{\mathbf{c}, \mathbf{c}} \|_{F}^2
      \]
   5. Classify the testing pixel $\mathbf{z}$:
      \[
      \text{Class}(\mathbf{z}) = \arg\min_{c=1,\ldots,C} r^c(\mathbf{z})
      \]

Here, $\mathbf{K}_{\mathbf{X}, \mathbf{Z}} \in \mathbb{R}^{N \times M_2}$ is the Log-Euclidean kernel matrix between the training samples and the neighboring pixels whose $(i, j)$th entry is $\kappa(\mathbf{C}_x, \mathbf{C}_z)$, and $\mathbf{K}_{\mathbf{X}, \mathbf{X}} \in \mathbb{R}^{N \times N}$ is the Log-Euclidean kernel matrix for training samples with $(i, j)$th entry $\kappa(\mathbf{C}_x, \mathbf{C}_x)$.

The pseudocode of LogEKJSR is shown in Algorithm 1.

### IV. EXPERIMENTS

In this section, we evaluate the effectiveness of our proposed LogEKJSR method on three benchmark HSIs (i.e., Indian Pines, Pavia University, and Salinas), and compare it with the following classification methods: support vector machine (SVM), SVM with composite kernel (SVM-CK) [7], JSR [15], NLW-JSR (WKJSR for simplicity) [16], and kernel-based JSR (KJSR) [26]. Because EMAPs and covariance features are used in the proposed method, we also compare the SVM on the EMAP data (SVM-EMAP) and SVM on the COVariance features extracted from the EMAP data (SVM-COV). Here, a Log-Euclidean linear kernel is used in the SVM-COV as recommended in [35]. In addition, a recently proposed deep learning-based hyperspectral classification method, i.e., spatial-spectral squeeze-and-excitation residual network (SSSERN) [36], is also compared. The class-specific accuracy, overall accuracy (OA) and $\kappa$ coefficient on the testing set are used for quantitative assessment. All data are normalized to have a unit $\ell_2$ norm. Except for SSSERN, all other experiments are carried using MATLAB R2017a and run in a computer with a 3.40 GHz Intel(R) Core(TM) i7-6700 processor. The SSSERN is implemented based on the TensorFlow deep learning framework of Python in the setting tensorflow-gpu-1.12, GeForce GTX 1080 Ti, and Intel(R) Xeon(R) CPU E5630.

For the proposed LogEKJSR, it needs to construct local regions for covariance feature computation and also needs to extract neighborhood pixel set for the joint sparse representation. In both the covariance feature computation and joint sparse representation, the neighborhood window sizes (i.e., $w_1$ and $w_2$) are set as 9, and the number of consistent and similar pixels in the neighborhoods are set as $M_1 = 70$ and $M_2 = 30$, respectively. The sparsity level in KJSR is set as: $K = 40$. The Gaussian Log-Euclidean kernel with parameter $\beta = 0.001$ is used in the proposed LogEKJSR. The above parameters are used for all the three data sets.

#### A. Indian Pines

This data has the size of $145 \times 145$ pixels and 220 spectral bands, and contains 16 different ground-truth classes. The number of samples in each class is unbalanced, ranging from 20 to 2468, which makes the classification of Indian Pines a challenging problem. The false color composite image and ground-truth map are shown in Fig. 4.

We randomly draw 5% labeled samples from each class to form the training set, and put all remaining labeled samples into the testing set. The algorithms are randomly run ten times and the averaged classification results of ten runs on the testing set are recorded, as shown in Table I. From the results, we can see that:

1. SVM-EMAP provides better results than SVM-CK, which demonstrates that EMAP features are more discriminative than mean features for the SVM classification. By further extracting the covariance features from EMAPs, SVM-EMAP slightly improves SVM-EMAP.
2. By extending JSR from the original space to the kernel space, KJSR dramatically improves JSR and WKJSR. It shows that the kernel method is effective to model the underlying nonlinear relation in the HSI.
3. Due to the limited training samples, the deep learning method SSSERN generates poor results on Classes 1, 7, and 16. Deep learning method usually needs a large number of training samples to train a good model. However, our proposed LogEKJSR is less dependent on the number of training samples and shows relatively better results on these three classes.
4. The proposed LogEKJSR provides the best results and the advantages are overwhelming. The Log-Euclidean kernel is better than the general kernel by comparing KJSR with LogEKJSR. Although both the LogEKJSR and SVM-EMAP use the Log-Euclidean kernels, the LogEKJSR is much more effective than SVM-EMAP.
On the subclasses of “Corn” (Classes 2, 3, 4) and “Soybean” (Classes 10, 11, 12), the proposed LogEKJSR provides higher classification accuracies than all other SVM-based and JSR-based methods. Because their spectral characteristics are highly similar, it is very difficult to classify them. LogEKJSR can effectively discriminate these spectral highly similar classes by exploiting both spectral correlation with covariance features and spatial correlation with EMAP features. For classes with very limited training samples, such as Classes 1, 4, 7, and 9, LogEKJSR also shows excellent results.

The running time for different methods when reaching their optimal classification performances are also reported in the Table I. The running time of LogEKJSR includes the total time for computing the EMAPs, covariance features, Log-Euclidean kernels, and kernel-based JSR. It can be seen that LogEKJSR takes only little more time than JSR. It should be noted that SSSERN runs in a different platform (i.e., Python) and the training process of SSSERN is time-consuming.

The classification maps of different algorithms on the entire image scene are shown in Fig. 5. Compared with the existing SVM and JSR methods, the proposed LogEKJSR improves the classification map not only on large homogeneous regions but also on boundary regions. Particularly, LogEKJSR can well preserve the borders between different land cover classes. In general, the classification map of LogEKJSR shows high consistency with little “salt & pepper” noise.

We further investigate the classification OA of different methods as the changes of the ratio of labeled samples per class for training (i.e., 1%, 3%, 5%, 7%, 9%) and show the results in

<table>
<thead>
<tr>
<th>Class</th>
<th>Train</th>
<th>Test</th>
<th>SVM</th>
<th>SVM-CK</th>
<th>SVM-EMAP</th>
<th>SVM-COV</th>
<th>SSSERN</th>
<th>JSR</th>
<th>WJSR</th>
<th>KJSR</th>
<th>LogEKJSR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>31</td>
<td>58.43</td>
<td>57.29</td>
<td>88.63</td>
<td>91.96</td>
<td>55.81</td>
<td>62.75</td>
<td>65.88</td>
<td>74.31</td>
<td>98.82</td>
</tr>
<tr>
<td>2</td>
<td>72</td>
<td>1362</td>
<td>79.23</td>
<td>90.26</td>
<td>88.57</td>
<td>91.76</td>
<td>93.07</td>
<td>86.90</td>
<td>88.63</td>
<td>91.16</td>
<td>96.48</td>
</tr>
<tr>
<td>3</td>
<td>42</td>
<td>792</td>
<td>67.85</td>
<td>90.66</td>
<td>91.55</td>
<td>93.17</td>
<td>92.89</td>
<td>76.62</td>
<td>87.95</td>
<td>95.57</td>
<td>95.59</td>
</tr>
<tr>
<td>4</td>
<td>12</td>
<td>222</td>
<td>60.18</td>
<td>80.04</td>
<td>76.80</td>
<td>87.39</td>
<td>100.0</td>
<td>80.09</td>
<td>81.58</td>
<td>86.26</td>
<td>95.59</td>
</tr>
<tr>
<td>5</td>
<td>25</td>
<td>472</td>
<td>90.87</td>
<td>92.16</td>
<td>93.37</td>
<td>91.29</td>
<td>93.01</td>
<td>86.61</td>
<td>88.69</td>
<td>91.39</td>
<td>94.11</td>
</tr>
<tr>
<td>6</td>
<td>37</td>
<td>710</td>
<td>95.22</td>
<td>97.08</td>
<td>97.00</td>
<td>95.48</td>
<td>100.0</td>
<td>98.82</td>
<td>99.39</td>
<td>99.13</td>
<td>97.79</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>23</td>
<td>80.43</td>
<td>84.35</td>
<td>88.07</td>
<td>90.00</td>
<td>84.62</td>
<td>86.69</td>
<td>86.96</td>
<td>86.96</td>
<td>96.09</td>
</tr>
<tr>
<td>8</td>
<td>24</td>
<td>465</td>
<td>96.92</td>
<td>95.38</td>
<td>99.48</td>
<td>96.82</td>
<td>100.0</td>
<td>99.53</td>
<td>99.81</td>
<td>99.91</td>
<td>99.78</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>17</td>
<td>84.71</td>
<td>90.00</td>
<td>95.88</td>
<td>97.06</td>
<td>100.0</td>
<td>23.53</td>
<td>40.00</td>
<td>4.12</td>
<td>99.41</td>
</tr>
<tr>
<td>10</td>
<td>48</td>
<td>920</td>
<td>70.58</td>
<td>84.14</td>
<td>87.53</td>
<td>90.02</td>
<td>92.52</td>
<td>73.13</td>
<td>74.73</td>
<td>87.61</td>
<td>94.23</td>
</tr>
<tr>
<td>11</td>
<td>123</td>
<td>2345</td>
<td>82.96</td>
<td>91.92</td>
<td>96.19</td>
<td>93.88</td>
<td>98.28</td>
<td>93.14</td>
<td>95.15</td>
<td>97.63</td>
<td>98.52</td>
</tr>
<tr>
<td>12</td>
<td>31</td>
<td>583</td>
<td>75.59</td>
<td>89.12</td>
<td>84.61</td>
<td>87.74</td>
<td>95.56</td>
<td>77.08</td>
<td>83.26</td>
<td>92.37</td>
<td>94.85</td>
</tr>
<tr>
<td>13</td>
<td>11</td>
<td>201</td>
<td>97.91</td>
<td>97.66</td>
<td>98.26</td>
<td>87.66</td>
<td>100.0</td>
<td>98.81</td>
<td>99.60</td>
<td>99.40</td>
<td>98.46</td>
</tr>
<tr>
<td>14</td>
<td>65</td>
<td>1229</td>
<td>95.57</td>
<td>96.43</td>
<td>99.25</td>
<td>99.24</td>
<td>96.25</td>
<td>98.44</td>
<td>99.20</td>
<td>99.00</td>
<td>99.84</td>
</tr>
<tr>
<td>15</td>
<td>19</td>
<td>361</td>
<td>52.82</td>
<td>82.10</td>
<td>92.94</td>
<td>96.04</td>
<td>93.99</td>
<td>87.23</td>
<td>84.63</td>
<td>84.54</td>
<td>98.20</td>
</tr>
<tr>
<td>16</td>
<td>5</td>
<td>90</td>
<td>90.67</td>
<td>96.11</td>
<td>95.78</td>
<td>93.22</td>
<td>88.64</td>
<td>93.33</td>
<td>99.11</td>
<td>91.33</td>
<td>96.78</td>
</tr>
<tr>
<td>Overall accuracy</td>
<td>81.76</td>
<td>91.23</td>
<td>93.12</td>
<td>93.38</td>
<td>95.72</td>
<td>88.46</td>
<td>90.20</td>
<td>93.70</td>
<td>94.57</td>
<td>97.25</td>
<td></td>
</tr>
<tr>
<td>Coefficient $\kappa$</td>
<td>79.16</td>
<td>90.00</td>
<td>92.15</td>
<td>92.46</td>
<td>95.12</td>
<td>86.79</td>
<td>88.77</td>
<td>92.80</td>
<td>96.86</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Running time (s)</td>
<td>0.30</td>
<td>0.94</td>
<td>7.42</td>
<td>14.43</td>
<td>318.7</td>
<td>135.8</td>
<td>143.7</td>
<td>106.1</td>
<td>170.8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5) On the subclasses of “Corn” (Classes 2, 3, 4) and “Soybean” (Classes 10, 11, 12), the proposed LogEKJSR provides higher classification accuracies than all other SVM-based and JSR-based methods. Because their spectral characteristics are highly similar, it is very difficult to classify them. LogEKJSR can effectively discriminate these spectral highly similar classes by exploiting both spectral correlation with covariance features and spatial correlation with EMAP features. For classes with very limited training samples, such as Classes 1, 4, 7, and 9, LogEKJSR also shows excellent results.
and standard deviation attributes (i.e., EAP differential morphological profile (DMP), EAP with area, inertia, extraction methods: extended morphological profile (EMP), differential Log-Euclidean kernels on different features. To this end, besides the EMAP, we also consider the following feature
down especially when enough training samples are available. The deep learning method SSSERN needs many training samples (i.e., 9% labeled samples) to achieve comparable performance with LogEKJSR. As the decrease of training samples, SSSERN suffers from a dramatically performance degradation. It is clear that the proposed LogEKJSR produces consistent better results than all other algorithms with different numbers of training samples.

In addition, we compare the performance of LogEKJSR with different Log-Euclidean kernels on different features. To this end, besides the EMAP, we also consider the following feature extraction methods: extended morphological profile (EMP), differential morphological profile (DMP), EAP with area, inertia, and standard deviation attributes (i.e., EAP, EAP, EAP). We perform the general KJSR on these features and further extract covariance matrix features on them to construct linear, polynomial, exponential, and Gaussian Log-Euclidean kernels for the LogEKJSR classification. The results of KJSR methods with different kernels on different features are shown in Table II. It can be seen that Log-Euclidean kernels show better results than the general kernel, the nonlinear Log-Euclidean kernels are generally more effective than the linear one, and three nonlinear Log-Euclidean kernels show similar results on this data set. As a combination of different EAPs, EMAP shows slightly better results than EAP, EAP, and EAP. Moreover, LogEKJSR on EMAP produces the best results.

The proposed LogEKJSR method includes several steps, such as extracting EMAP features, computing the covariance matrix, calculating the Log-Euclidean kernel, and executing kernel-based JSR classification. Now, we analyze the contribution of each step. We consider the following three situations: 1) No EMAP, which computes the covariance matrix on the original high-dimensional hyperspectral data rather than EMAP data and then performs Log-Euclidean kernel-based JSR classification; 2) No COV, which performs kernel-based JSR classification directly on the EMAP data; and 3) No LogE kernel, which extracts EMAP features and compute the covariance matrices, and then perform the SVM classification rather than KJSR. The corresponding OA results are shown in Table III, from which the following conclusion can be drawn.

1) The OA of LogEKJSR decreases nearly 1% when the EMAP is not used. Note that the computation of covariance matrices on the original high-dimensional hyperspectral data is time-consuming, and the total time corresponding to the “No EMAP” case is 3445 s. The EMAP operation can not only improve the classification accuracy but also reduce the computational time.

2) Without the COV features, the OA is only 95.51%. The result demonstrates that the covariance features and covariance-matrix-based Log-Euclidean kernel are effective for HSI classification.

3) In the “No LogE kernel” case, by replacing the KJSR with SVM, the OA is decreased from 97.25% to 93.38%. It demonstrates that the Log-Euclidean kernel-based KJSR model is a powerful classifier.

4) When the EMAP features, covariance matrices, Log-Euclidean kernel, and KJSR are used, the proposed LogEKJSR generates the best result.

In order to further explain the effectiveness of the proposed LogEKJSR, we construct a simulation example to show that the covariance-based matrix-to-matrix similarity (LogE-Gaussian kernel) is more accurate than the traditional Gaussian-kernel-based vector-to-vector similarity. We select a pixel from the “Soybean-notill” class as the target sample, and find five other samples from the same class and five from a different class (i.e., “Soybean-mintill” class). The spectral curve of these samples is shown in Fig. 7. It can be clearly seen that the target sample is more similar to the five inhomogeneous samples than the five homogeneous ones.

Table IV shows within-class similarities between the target sample and the five samples from the “Soybean-notill” class, and between-class similarities between the target sample and the five samples from the “Soybean-mintill” class, where the similarity is measured by the Gaussian and Log-Euclidean Gaussian kernels, respectively. It is obvious that the Gaussian-kernel-based within-class similarities are smaller than the between-class similarities.
This means that the target sample is more similar to the samples from a different class rather than the same class, which coincides with the visual result in Fig. 7. Because the “Soybean-notill” and “Soybean-mintill” are subclasses of “Soybean,” their spectral characteristics are very similar. Due to spectral variation and intrinsic material similarity, it is difficult to discriminate these two classes by the Gaussian-kernel-based method. Although the spectral curve of the target sample is similar to that of samples from “Soybean-mintill,” the spatial structure of these samples are usually different. By exploiting the spatial structure information in form of covariance matrix and measuring the matrix-to-matrix similarity directly, the LogE-Gaussian kernel increases the within-class similarities and reduces the between-class similarities, which increases the class separability. Now, within-class similarities are larger than the between-class similarities, and the target sample can be correctly classified. Due to the effectiveness of Log-Euclidean kernel, the proposed LogEKJSR can produce accurate classification.

The effect of parameters on the final classification result is analyzed as follows.

1) The effect of region parameters $M_1$ and $M_2$: The construction of local regions for covariance feature computation and the extraction of similar neighboring pixels for the joint sparse representation are the two key factors in the proposed LogEKJSR. The local region for the covariance feature extraction of each HSI pixel is obtained by selecting $M_1$ consistent pixels from a $w_1 \times w_1$ window, and the neighboring pixel set used in the joint sparse representation of each testing pixel consists of $M_2$ most similar pixels extracted from a $w_2 \times w_2$ window. Because similar pixels are selected from the $w_1 \times w_1$ and $w_2 \times w_2$ windows, a large window can be chosen in the first stage. In the experiments, $w_1$ and $w_2$ are set as 9. We investigate the effect of $M_1$ and $M_2$, and show the corresponding OAs of LogEKJSR in Fig. 8. From Fig. 8(a), $M_1 = 70$ corresponds to the best result, which indicates that eliminating some inconsistent pixels inside the spatial window for the covariance descriptor slightly benefits the classification results. On the one hand, more neighboring pixels should be used to capture the rich spatial information. On the other hand, not all neighboring pixels are useful. So, a relatively large value of $M_1 = 70$ is used. From Fig. 8(b), the performance of LogEKJSR slightly improves as $M_2$ increases and tends to saturate when $M_2$ is between 20 and 50. The results demonstrate that the pixels in the neighborhood of a testing pixel should be carefully chosen such that the neighborhood similarity or joint sparsity assumption is met for joint sparse representation. It is sufficient to choose half of pixels in the $9 \times 9$ neighborhood that are similar to the central pixel.

2) The effect of Gaussian kernel parameter $\beta$: For the Gaussian Log-Euclidean kernel, there is a parameter $\beta$ related to the width of Gaussian function. Fig. 9 shows the changes of OA of the proposed LogEKJSR as a function of $\beta$, where the parameter $\beta$ varies in the range $\{0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5\}$. It can be seen that LogEKJSR method achieves the best result at $\beta = 0.001$.

3) The effect of sparsity level $K$: Fig. 10 shows the effect of sparsity level $K$ on the proposed LogEKJSR, where the parameter $K$ varies in the range $\{10, 20, 30, 40, 50, 60, 70, 80\}$. It can be seen that LogEKJSR show relatively stable results when $K$ is greater than 30.
TABLE V

CLASSIFICATION RESULTS FOR THE PAVIA UNIVERSITY DATA SET (1% LABELED SAMPLES PER CLASS FOR TRAINING, A TOTAL OF 426 TRAINING SAMPLES, AND 42350 TESTING SAMPLES)

<table>
<thead>
<tr>
<th>Class</th>
<th>Train</th>
<th>Test</th>
<th>SVM</th>
<th>SVM-CK</th>
<th>SVM-EMAP</th>
<th>SVM-COV</th>
<th>SSSERN</th>
<th>JSR</th>
<th>WJSR</th>
<th>KJSR</th>
<th>LogEKJSR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>66</td>
<td>6565</td>
<td>83.51</td>
<td>94.08</td>
<td>98.35</td>
<td>99.41</td>
<td>98.25</td>
<td>61.08</td>
<td>61.89</td>
<td>80.25</td>
<td>99.85</td>
</tr>
<tr>
<td>2</td>
<td>186</td>
<td>18463</td>
<td>96.02</td>
<td>98.58</td>
<td>99.45</td>
<td>99.25</td>
<td>99.94</td>
<td>98.33</td>
<td>98.33</td>
<td>98.24</td>
<td>99.56</td>
</tr>
<tr>
<td>3</td>
<td>21</td>
<td>2078</td>
<td>60.72</td>
<td>80.11</td>
<td>97.08</td>
<td>89.98</td>
<td>92.40</td>
<td>91.53</td>
<td>71.63</td>
<td>71.47</td>
<td>67.10</td>
</tr>
<tr>
<td>4</td>
<td>31</td>
<td>3033</td>
<td>85.21</td>
<td>92.10</td>
<td>96.22</td>
<td>99.90</td>
<td>94.39</td>
<td>76.40</td>
<td>78.85</td>
<td>78.40</td>
<td>98.49</td>
</tr>
<tr>
<td>5</td>
<td>13</td>
<td>1332</td>
<td>99.02</td>
<td>99.90</td>
<td>99.32</td>
<td>97.63</td>
<td>100.00</td>
<td>99.91</td>
<td>99.65</td>
<td>99.87</td>
<td>99.57</td>
</tr>
<tr>
<td>6</td>
<td>50</td>
<td>4979</td>
<td>58.29</td>
<td>85.22</td>
<td>97.71</td>
<td>98.47</td>
<td>94.42</td>
<td>72.49</td>
<td>67.39</td>
<td>74.63</td>
<td>99.56</td>
</tr>
<tr>
<td>7</td>
<td>13</td>
<td>1317</td>
<td>69.63</td>
<td>93.45</td>
<td>99.79</td>
<td>98.08</td>
<td>97.95</td>
<td>92.27</td>
<td>92.47</td>
<td>62.89</td>
<td>99.93</td>
</tr>
<tr>
<td>8</td>
<td>37</td>
<td>3645</td>
<td>81.41</td>
<td>87.05</td>
<td>97.16</td>
<td>97.52</td>
<td>97.67</td>
<td>90.70</td>
<td>89.57</td>
<td>76.96</td>
<td>98.57</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>938</td>
<td>95.11</td>
<td>92.77</td>
<td>99.94</td>
<td>97.18</td>
<td>100.00</td>
<td>68.23</td>
<td>89.24</td>
<td>78.03</td>
<td>95.10</td>
</tr>
</tbody>
</table>

Overall accuracy | 85.14 | 93.70 | 98.58 | 98.43 | 98.01 | 85.17 | 85.24 | 86.40 | 99.06 |
Coefficient κ    | 79.97 | 91.60 | 98.12 | 97.93 | 97.35 | 80.13 | 80.15 | 81.68 | 98.75 |
Running time (s) | 0.62  | 2.01  | 61.09 | 96.99 | 241.9 | 122.0 | 135.8 | 200.6 | 490.6 |

Fig. 10. Classification OA versus sparsity level $K$.

Fig. 11. Pavia University data set. (a) RGB composite image of three bands 60, 30, and 2. (b) Ground-truth map.

Based on the above analysis, we set $M_1 = 70$, $M_2 = 30$, $\beta = 0.001$, $K = 40$ in the experiments and use the same parameters for all three data sets for consistency.

B. Pavia University

This data was acquired in 2001 by the reflective optics system imaging spectrometer (ROSIS) instrument over the city of Pavia, Italy. This imaging scene corresponds to the University of Pavia and has the size of 610 × 340 pixels and 115 spectral bands. After discarding noisy and water absorption bands, 103 bands are retained. The data contains nine ground-truth classes. The false color composition image and the ground-truth map are shown in Fig. 11.

For this data, we randomly select 1% labeled samples from each class to form the training set. The remaining samples consist of the testing set. The averaged classification results over ten runs are recorded in Table V. It can be seen from the results that traditional sparse-based spatial-spectral methods (i.e., JSR and WJSR) even show worse results than spectral-based SVM in the case with only 1% labeled samples for training. When the number of training samples is limited, the representation ability of the training dictionary is poor, which leads to a lower discriminative performance of sparse-based methods. Notwithstanding, by extracting the covariance features on the EMAPs and using a Log-Euclidean kernel, the proposed LogEKJSR model produces excellent results. Compared with the JSR and KJSR, LogEKJSR improves the OA by about 13%, and the $\kappa$ accuracy by about

TABLE VI

COMPARISON OF KJSR METHODS WITH DIFFERENT KERNELS ON DIFFERENT FEATURES FOR PAVIAU

<table>
<thead>
<tr>
<th></th>
<th>KJSR</th>
<th>LogEKJSR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Labeled</td>
<td>Linear</td>
<td>Poly</td>
</tr>
<tr>
<td>EMP</td>
<td>94.51</td>
<td>92.58</td>
</tr>
<tr>
<td>DMP</td>
<td>83.89</td>
<td>87.43</td>
</tr>
<tr>
<td>EAP_a</td>
<td>96.94</td>
<td>97.54</td>
</tr>
<tr>
<td>EAP_p</td>
<td>97.22</td>
<td>97.17</td>
</tr>
<tr>
<td>EAP_s</td>
<td>95.10</td>
<td>97.41</td>
</tr>
<tr>
<td>EMAP</td>
<td>97.11</td>
<td>98.62</td>
</tr>
</tbody>
</table>
TABLE VII

<table>
<thead>
<tr>
<th>Class</th>
<th>Tram</th>
<th>Test</th>
<th>SVM</th>
<th>SVM-CK</th>
<th>SVM-EMAP</th>
<th>SVM-COV</th>
<th>SSSERN</th>
<th>JSR</th>
<th>WJSR</th>
<th>KJSR</th>
<th>LogEKJSR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td>1989</td>
<td>97.90</td>
<td>99.31</td>
<td>100.0</td>
<td>99.70</td>
<td>99.97</td>
<td>99.95</td>
<td>99.98</td>
<td>100.0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>37</td>
<td>3689</td>
<td>98.76</td>
<td>99.64</td>
<td>99.61</td>
<td>99.89</td>
<td>100.0</td>
<td>99.92</td>
<td>99.94</td>
<td>99.89</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>1956</td>
<td>98.45</td>
<td>97.38</td>
<td>99.94</td>
<td>98.52</td>
<td>98.64</td>
<td>91.33</td>
<td>91.76</td>
<td>95.79</td>
<td>99.89</td>
</tr>
<tr>
<td>4</td>
<td>14</td>
<td>1380</td>
<td>99.28</td>
<td>97.43</td>
<td>99.16</td>
<td>90.26</td>
<td>98.91</td>
<td>98.07</td>
<td>98.40</td>
<td>97.74</td>
<td>97.59</td>
</tr>
<tr>
<td>5</td>
<td>27</td>
<td>2651</td>
<td>97.40</td>
<td>97.90</td>
<td>98.45</td>
<td>95.79</td>
<td>98.30</td>
<td>85.76</td>
<td>90.01</td>
<td>97.98</td>
<td>98.04</td>
</tr>
<tr>
<td>6</td>
<td>40</td>
<td>3919</td>
<td>99.31</td>
<td>99.79</td>
<td>99.68</td>
<td>99.49</td>
<td>99.40</td>
<td>100.0</td>
<td>99.87</td>
<td>99.92</td>
<td>99.98</td>
</tr>
<tr>
<td>7</td>
<td>36</td>
<td>3543</td>
<td>99.38</td>
<td>98.83</td>
<td>97.96</td>
<td>97.76</td>
<td>91.19</td>
<td>99.93</td>
<td>99.89</td>
<td>99.86</td>
<td>99.98</td>
</tr>
<tr>
<td>8</td>
<td>113</td>
<td>11158</td>
<td>89.64</td>
<td>92.27</td>
<td>91.28</td>
<td>98.94</td>
<td>93.28</td>
<td>90.76</td>
<td>91.03</td>
<td>92.02</td>
<td>99.66</td>
</tr>
<tr>
<td>9</td>
<td>62</td>
<td>6141</td>
<td>97.51</td>
<td>99.06</td>
<td>99.36</td>
<td>99.81</td>
<td>99.98</td>
<td>99.95</td>
<td>99.96</td>
<td>100.0</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>33</td>
<td>3245</td>
<td>88.47</td>
<td>94.03</td>
<td>94.90</td>
<td>97.58</td>
<td>99.26</td>
<td>92.48</td>
<td>92.72</td>
<td>95.46</td>
<td>99.16</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>1057</td>
<td>93.68</td>
<td>94.53</td>
<td>95.65</td>
<td>99.58</td>
<td>95.37</td>
<td>94.42</td>
<td>93.98</td>
<td>90.08</td>
<td>99.41</td>
</tr>
<tr>
<td>12</td>
<td>19</td>
<td>1908</td>
<td>99.91</td>
<td>99.62</td>
<td>100.0</td>
<td>99.35</td>
<td>100.0</td>
<td>79.07</td>
<td>82.41</td>
<td>99.74</td>
<td>99.94</td>
</tr>
<tr>
<td>13</td>
<td>9</td>
<td>907</td>
<td>97.10</td>
<td>96.29</td>
<td>98.10</td>
<td>95.59</td>
<td>98.00</td>
<td>97.89</td>
<td>89.08</td>
<td>98.30</td>
<td>98.63</td>
</tr>
<tr>
<td>14</td>
<td>11</td>
<td>1059</td>
<td>93.09</td>
<td>96.77</td>
<td>96.07</td>
<td>96.77</td>
<td>98.77</td>
<td>95.49</td>
<td>95.33</td>
<td>98.75</td>
<td>98.60</td>
</tr>
<tr>
<td>15</td>
<td>73</td>
<td>7195</td>
<td>56.57</td>
<td>81.44</td>
<td>93.36</td>
<td>94.73</td>
<td>55.71</td>
<td>54.31</td>
<td>54.48</td>
<td>64.34</td>
<td>97.54</td>
</tr>
<tr>
<td>16</td>
<td>18</td>
<td>1789</td>
<td>96.05</td>
<td>95.26</td>
<td>99.91</td>
<td>99.12</td>
<td>98.82</td>
<td>99.05</td>
<td>99.04</td>
<td>99.16</td>
<td>99.93</td>
</tr>
</tbody>
</table>

Overall accuracy 89.77 94.61 96.57 98.13 91.40 89.07 89.66 92.65 99.36
Coefficient $r$ 88.57 94.00 96.18 97.92 90.39 87.78 88.45 91.80 99.28
Running time (s) 1.11 7.66 20.09 38.82 352.5 503.1 545.6 328.3 399.5

Fig. 13. (a) False color composite image and (b) ground-truth map of Salinas.

Fig. 14. Classification OA versus the ratio of training samples per class for Salinas.

18%. Compared with SVM-EMAP and SVM-COV, LogEKJSR also provides relatively better results.

The classification OA of different methods as the changes of the ratio of training samples per class (i.e., 0.2%, 0.4%, 0.6%, 0.8%, 1%) are shown in Fig. 12. The proposed LogEKJSR shows consistent better results than all other algorithms.

The results of KJSR methods with different kernels on different features are shown in Table VI. Similar to results on Indian Pines, EMAP provides the best results on different kernels. By merging EAPs, EMAP improves EAP$_a$, EAP$_i$ and EAP$_s$. In addition, the nonlinear Log-Euclidean kernels show slightly better results than the linear one.

C. Salinas

This image has the size of $512 \times 217$ pixels and 204 spectral bands. The data contains 16 ground-truth classes and totally 54129 labeled samples. The false color composite image and the ground-truth map are shown in Fig. 13.

For this data, we randomly select 1% labeled samples from each class to form the training set. The remaining samples consist of the testing set. The averaged classification results over ten runs are recorded in Table VII. It is clear that the proposed LogEKJSR shows the best classification performance, and improves the OA of KJSR and JSR by more than 6% and 10%, respectively, which demonstrates the effectiveness of Log-Euclidean kernel JSR method over the general Gaussian kernel JSR and linear JSR. In particular, on the subclasses of “Lettuce” (i.e., Classes 12, 13, 14), kernel-based JSR methods (i.e., KJSR and LogEKJSR) dramatically improve the original JSR and WJSR methods. We can also see that almost all traditional sparse-based methods, SVM methods, and deep learning method SSSERN show bad results on Classes 8 and 15 (i.e., “Grapes untrained” and “Vinyard untrained”). These two classes with majority samples are located on the upper left of the image and are spatially adjacent. Due to the spectral similarity, these two classes are very difficult to be distinguished from each other. Notwithstanding, our proposed LogEKJSR almost correctly classifies all the samples in these two classes.

The OA of different methods as the changes of the ratio of training samples per class (i.e., 0.2%, 0.4%, 0.6%, 0.8%, 1%) are shown in Fig. 14. The proposed LogEKJSR shows consistent better results than all other algorithms with different
numbers of training samples. Here, the number of training samples is relatively small, all the existing JSR methods, KJSR and SSSERN offer poor results. SVM-COV produces better results than SVM-EMAP. It also shows that the covariance features extracted from EMAPs is more discriminative than the original EMAPs.

The results of KJSR methods with different kernels on different features are shown in Table VIII. By merging EAPs, EMAP improves $\text{EAP}_a$, $\text{EAP}_1$ and $\text{EAP}_s$ and provides the best results.

V. Conclusion

In this article, we have proposed a LogEJKSR model for classification of HSIs. It first extracts EMAP features, and then computes local covariance matrices on the EMAPs. Based on the Log-Euclidean distance on the Riemannian manifold, the distance between covariance matrices can be accurately computed and the traditional kernels can be generalized to Log-Euclidean kernels. Finally, a LogEJKSR model is obtained by replacing the traditional kernel in the KJSR with the Log-Euclidean kernel. Experimental results on three benchmark hyperspectral data sets have demonstrated that the proposed LogEJKSR method is much more effective than existing JSR and SVM methods. The proposed LogEJKSR extracts covariance features from a square spatial neighborhood and uses traditional Gaussian type kernel to measure the similarity between covariance matrices. In the future, we can consider to construct adaptive local regions by means of superpixel segmentation methods and employ multiple kernel learning method to learn an appropriate kernel.

Acknowledgment

The authors would like to thank Prof. D. Landgrebe for providing the Indian Pines data set, Prof. P. Gamba for providing the University of Pavia data set, and Dr. J. Anthony Gualtieri for providing Salinas data set. Additionally, they also thank Mauro Dalla Mura for sharing the EMAP code.

References


Weidong Yang received the B.S. degree in industrial electrification and automation from the Huazhong University of Science and Technology in 1997, the M.S. and Ph.D. degrees in pattern recognition and intelligent systems from the Huazhong University of Science and Technology, Wuhan, China, in 2000 and 2005, respectively. Currently, he is with the School of Artificial Intelligence and Automation, Huazhong University of Science and Technology. His research interests include pattern recognition, computer vision, multispectral image processing in remote sensing and autonomous navigation.

Jiangtao Peng received the B.S. and M.S. degrees from Hubei University, Wuhan, China, in 2005 and 2008, and the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2011. He is currently a Professor with the Faculty of Mathematics and Statistics, Hubei University, Wuhan, China. His research interests include machine learning and hyperspectral image processing.

Weiwei Sun (M’15) received the B.S. degree in surveying and mapping from Tongji University in 2007, and Ph.D. degree in cartography and geographic information engineering from Tongji University, Shanghai, China, in 2013. He is currently an Associate Professor with Ningbo University, Beijing, China, and. He has authored or coauthored more than 70 journal papers and his current research interests include hyperspectral image processing with manifold learning, anomaly detection, and target recognition of remote sensing imagery using compressive sensing.

Qian Du (S’98–M’00–SM’05–F’18) received the Ph.D. degree in electrical engineering from the University of Maryland, Baltimore County, Baltimore, MD, USA, in 2000. She is currently a Bobby Shackouls Professor with the Department of Electrical and Computer Engineering, Mississippi State University, Mississippi State, MS, USA. Her research interests include hyperspectral remote sensing image analysis and applications, pattern classification, data compression, and neural networks. Dr. Du is a fellow of SPIE-International Society for Optics and Photonics and a fellow of IEEE. She was a recipient of the 2010 Best Reviewer Award from the IEEE Geoscience and Remote Sensing Society (GRSS). She served as a Cochair for the Data Fusion Technical Committee of the IEEE GRSS from 2009 to 2013. She was the Chair with the Remote Sensing and Mapping Technical Committee of International Association for Pattern Recognition from 2010 to 2014. She was the General Chair for the fourth IEEE GRSS Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing held at Shanghai, China, in 2012. She served as an Associate Editor for the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING (JSTARS), the Journal of Applied Remote Sensing, and the IEEE SIGNAL PROCESSING LETTERS. Since 2016, she has been the Editor-in-Chief for the IEEE JSTARS.