Local-Manifold-Learning-Based Graph Construction for Semisupervised Hyperspectral Image Classification

Li Ma, *Member, IEEE*, Melba M. Crawford, *Fellow, IEEE*, Xiaoquan Yang, *Member, IEEE*, and Yan Guo

***Abstract*—Graph construction, which is at the heart of graph-based semisupervised learning (SSL), is investigated by using manifold learning (ML) approaches. Since each ML method can be demonstrated to correspond to a specific graph, we build the relation between ML and SSL via the graph, where ML methods are employed for graph construction. Moreover, sparsity is important for the efficiency of SSL algorithms, and therefore, local ML (LML)-method-based sparse graphs are utilized. The LML-based graphs are able to capture the local geometric properties of hyperspectral data and, thus, are beneficial for classification of data with complex geometry and multiple submanifolds. In experiments with Hyperion and AVIRIS hyperspectral data, graphs constructed by two LML methods, namely, locally linear embedding and local tangent space alignment (LTSA), performed better than several popular graph construction methods, and the highest accuracies were obtained by using graphs provided by LTSA.**

***Index Terms*—Graph, hyperspectral images, manifold learning (ML), semisupervised learning (SSL).**

I. Introduction

FOR hyperspectral image classification, supervised classifiers require a large quantity of labeled data due to the high-dimensional spectral data vector. However, labeled instances are often difficult, costly, or time consuming to obtain. Since unlabeled data are relatively easy to collect, semisupervised learning (SSL) that utilizes both labeled and unlabeled data is widely employed to solve the small-size sample problem [1]–[4].

Among SSL methods, graph-based approaches have recently received significant attention due to their efficiency and computational simplicity [5]–[9]. These methods utilize a graph Laplacian regularizer to constrain the classification function to be smooth with respect to the data manifold, a low-dimensional subspace on which the high-dimensional data actually reside; in SSL, the manifold is represented by a graph constructed from both labeled and unlabeled data. The graph construction step is critical in graph-based SSL but is still an open issue that has not been comprehensively studied [10], [11]. The process generally involves two steps: determination of graph adjacency relationships and graph edge weight calculation. In this paper, we focus on the latter as our goal is to investigate the relation between SSL and manifold learning (ML), which can characterize the nonlinear relationships between data points. For graph adjacency construction, k-nearest neighbors (kNN) and ε-ball neighborhood are the most commonly used methods to obtain a sparse graph. For edge weight calculation, most graph-based methods adopt a heat kernel function to measure the similarity between data points [5]–[9]; this method only considers the pairwise similarities between two data points, ignoring the neighborhood relationships. In recent years, some other graph weighting methods have been proposed. Wang and Zhang [12] employed nonnegative local linear reconstruction (LLR) coefficients as the edge weights, exploiting the geometry of each neighborhood. Both Yan and Wang [13] and Cheng *et al.* [14] proposed sparse representation (SR)-based graph weights, which encode the discriminative information, obtaining the graph adjacency structure and edge weights simultaneously. Because of the high computational cost of SR, Li *et al.* performed the SR locally in each neighborhood [15]. For hyperspectral data classification, Gu and Feng used SR to obtain the graph structure [16], and Camps-Valls *et al.* introduced composite kernels that combine spectral and spatial information [1] to set the graph weights. In this paper, we explore the potential value of ML methods for graph construction in graph-based SSL, since each ML method can be demonstrated to correspond to a specific graph [17]. Thus, by building the relations between ML and SSL through graph, the applications of ML are extended.

ML methods, which are proposed for nonlinear dimensionality reduction, have been successfully applied to hyperspectral remote sensing applications, including feature extraction [18]–[20], classification [21]–[25], and target detection [26], [27]. Popular ML methods are often categorized as global techniques such as isometric feature mapping (Isomap) [28] and local techniques such as locally linear embedding (LLE) [29], local tangent space alignment (LTSA) [30], and Laplacian eigenmaps (LE) [31]. In the graph embedding framework [17], manifold coordinates are obtained from the eigenvectors of a matrix that can be demonstrated to be the graph Laplacian matrix. Therefore,

Manuscript received April 17, 2014; revised September 5, 2014; accepted October 19, 2014. This work was supported in part by the National Natural Science Foundation of China under Grant 61102104 and Grant 81201067, by the Fundamental Research Funds for National University, China University of Geosciences (Wuhan) under Grant CUG120408 and Grant CUG120119, and by the Purdue Laboratory for Applications of Remote Sensing.

L. Ma is with the Faculty of Mechanical and Electronic Information, China University of Geosciences, Wuhan 430074, China (e-mail: maryparisster@gmail.com).

M. M. Crawford is with the School of Civil Engineering, Purdue University, West Lafayette, IN 47907 USA (e-mail: mcrawford@purdue.edu).

X. Yang is with the Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: yangxiaoquan@gmail.com).

Y. Guo is with the College of Computer Science, China University of Geosciences, Wuhan 430074, China (e-mail: 323110966@qq.com).

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

Digital Object Identifier 10.1109/TGRS.2014.2365676

0196-2892 © 2014 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

a graph that corresponds to the Laplacian matrix can be determined, and different ML methods correspond to different graph structures. In addition, local ML (LML) methods characterize the local geometry in each neighborhood, and therefore, only points included in a neighborhood are connected, which results in sparse graphs. Since sparsity is important for the efficiency of SSL algorithms [10], LML-method-based sparse graphs are utilized in this paper. In the LML-based graphs, the weights between two points are assumed to capture the geometric properties of their related neighborhoods, which is beneficial for data sets with complex geometry and multiple submanifolds [23]. Moreover, it can be demonstrated that LLE- and LTSA-based graph Laplacian regularizers constrain the prediction vectors to preserve the local geometry of each neighborhood, so the resultant SSL can leverage the good properties of LML methods. With the derived graph weighting formula of LML methods, an out-of-sample extension can also be implemented.

The contributions of this paper include the following: 1) By building the relations between LML and SSL via graphs, a new family of graphs based on LML is introduced to enrich the strategies for addressing classification with limited training data. LML methods have been demonstrated to be particularly effective for classification of hyperspectral data that exhibit intrinsic nonlinear properties [23], [32], [33]. The potential value of LML for graph-based SSL is explored in this context. 2) LTSA-based graph development is obtained by reformulating this method in the graph embedding framework.

The organization of this paper is as follows. Section II provides a brief introduction to graph-based SSL. Section III describes the LML-based graph construction approach. The out-of-sample extension is presented in Section IV. Experimental results are discussed in Section V, and conclusions are summarized in Section VI.

II. Graph-Based SSL

Popular graph-based SSL methods include Gaussian fields and harmonic functions (GFHF) [6], label propagation [7], Tikhonov regularization [8], and local and global consistency methods [9]. The objective function of these methods imposes a tradeoff between the empirical risk on labeled data points and smoothness of predictions on the graph, where the smoothness is measured by a graph Laplacian regularization term.

Suppose we have labeled data $\mathbf{X}_l = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_l]$ and unlabeled data $\mathbf{X}_u = [\mathbf{x}_{l+1}, \mathbf{x}_{l+2}, \ldots, \mathbf{x}_{l+u}]$; there are C classes in the data denoted as $\mathbf{\Omega} = [\Omega_1, \ldots, \Omega_C]$. The class labels of $\mathbf{X}_l$ are denoted as $\mathbf{Y}_l \in \mathrm{R}^{C\times l}$ with $\mathbf{Y}_{ij} = 1$ if $\mathbf{x}_j \in \Omega_i$. The data points $\mathbf{X} = [\mathbf{X}_l, \mathbf{X}_u]$ produce a weighted graph $G = (\mathbf{X}, \mathbf{W})$, where $\mathbf{X}$ consists of $N = l + u$ data points, and $\mathbf{W} \in \mathrm{R}^{N\times N}$ is the adjacency matrix of the connected edges between data points. For a directed graph, we use a simple method to obtain a symmetric adjacency matrix $\mathbf{W} = \mathbf{W} + \mathbf{W}^{\mathrm{T}}$. The graph Laplacian matrix is obtained by $\mathbf{L} = \mathbf{D} - \mathbf{W}$, where $\mathbf{D}$ is the diagonal degree matrix given by $\mathbf{D}_{ii} = \sum_j \mathbf{W}_{ij}$. Another widely employed normalized Laplacian matrix is denoted as $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$. The purpose of SSL is to predict the class of unlabeled data based on the whole data set $\mathbf{X}$ and the label information $\mathbf{Y}_l$.

The graph Laplacian regularization term is expressed by

$$\frac{1}{2}\sum_{i,j=1}^{N} \mathbf{W}_{ij}\|\mathbf{f}_i - \mathbf{f}_j\|^2 = \mathrm{Tr}(\mathbf{F}\mathbf{L}\mathbf{F}^{\mathrm{T}}) \tag{1}$$

where $\mathbf{F} \in \mathrm{R}^{C\times N}$ indicates the prediction vectors of data $\mathbf{X}$, and $\mathbf{f}_i \in \mathrm{R}^C$ and $\mathbf{f}_j \in \mathrm{R}^C$ are the predictions of points $\mathbf{x}_i$ and $\mathbf{x}_j$.

Since different graph-based SSL methods share the same regularization term, we only apply LML-based graphs to GFHF in this paper, although it can also be applied to other methods. GFHF constrains the predictions of labeled data to be equal to the label information $\mathbf{F}_l = \mathbf{Y}_l$, and the following optimization problem is solved:

$$\min_{\mathbf{F}} \mathrm{Tr}(\mathbf{F}\mathbf{L}\mathbf{F}^{\mathrm{T}}) \quad \text{s.t. } \mathbf{F}_l = \mathbf{Y}_l. \tag{2}$$

Since $\mathbf{F}_l$ is fixed, $\mathbf{L}$ is split into four blocks, and (2) becomes

$$\min_{\mathbf{F}_u}[\mathbf{Y}_l \quad \mathbf{F}_u]\begin{bmatrix}\mathbf{L}_{ll} & \mathbf{L}_{lu}\\ \mathbf{L}_{ul} & \mathbf{L}_{uu}\end{bmatrix}\begin{bmatrix}\mathbf{Y}_l^{\mathrm{T}}\\ \mathbf{F}_u^{\mathrm{T}}\end{bmatrix}. \tag{3}$$

Setting the derivative to zero with respect to $\mathbf{F}_u$, we obtain the solution

$$\mathbf{F}_u^{\mathrm{T}} = -\mathbf{L}_{uu}^{-1}\mathbf{L}_{ul}\mathbf{Y}_l^{\mathrm{T}}. \tag{4}$$

The predicted label of each unlabeled data point is determined by

$$y_i = \arg\max_{j=1,\ldots,C} f_{ji}, \quad i = 1, \ldots, u. \tag{5}$$

The predictions $\mathbf{F}_u$ rely heavily on the graph Laplacian matrix $\mathbf{L}$ or, equivalently, the edge weight matrix $\mathbf{W}$. For edge weighting, there are four popular weighting methods.

1) HK Function [5]–[9]:

$$\mathbf{W}_{ij} = \begin{cases} e^{-\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma}, & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are neighbors}\\ 0, & \text{otherwise}\end{cases} \tag{6}$$

where σ is the heat kernel (HK) parameter. The HK is the most commonly used graph edge weighting function.

2) LLR-Based Weight [12]: For each data point $\mathbf{x}_i, i = 1, 2, \ldots, N$, supposing its neighborhood that contains the k-nearest neighbors is denoted as $N(\mathbf{x}_i)$, the edge weights between $\mathbf{x}_i$ and its neighbors are represented by the linear coefficients that best reconstruct $\mathbf{x}_i$ from its neighbors with both sum-to-one and nonnegative constraints. The objective function is

$$\begin{aligned}\mathbf{W}_{ij} &= \arg\min \left\| \mathbf{x}_i - \sum_{\mathbf{x}_j \in N(\mathbf{x}_i)} \mathbf{W}_{ij}\mathbf{x}_j \right\|^2\\ &\text{s.t.} \sum_{\mathbf{x}_j \in N(\mathbf{x}_i)} \mathbf{W}_{ij} = 1, \quad \mathbf{W}_{ij} > 0.\end{aligned} \tag{7}$$

3) SR-Based Weight [13], [14]: Unlike HK- and LLR-based graphs that employ the kNN method to construct the adjacency graph, SR can obtain the graph adjacency structure and edge weights simultaneously. For each data point $\mathbf{x}_i$ that has been normalized to have unit l_2-norm, SR decomposes $\mathbf{x}_i$ as a sparse linear combination of the rest of the data points, i.e.,

$$\boldsymbol{\alpha}_i = \arg\min \|\boldsymbol{\alpha}_i\|_1, \quad \text{s.t.} \quad \mathbf{x}_i = \mathbf{D}_i\boldsymbol{\alpha}_i, \quad \boldsymbol{\alpha}_i \geq 0 \tag{8}$$

where $\boldsymbol{\alpha}_i$ is the nonnegative coefficient vector that best reconstructs $\mathbf{x}_i$ from $\mathbf{D}_i$ using l_1 optimization. The matrix

$\mathbf{D}_i = \{\mathbf{x}|\mathbf{x} \in \mathbf{X}, \mathbf{x} \neq \mathbf{x}_i\}$ is composed of all the normalized data points excluding $\mathbf{x}_i$; it is called "dictionary," which is used to represent the data point $\mathbf{x}_i$. An edge connects $\mathbf{x}_j$ and $\mathbf{x}_i$ if $\alpha_{ij} \neq 0$. The graph edge weight $\mathbf{W}_{ij} = \alpha_{ij}$.

4) LSR-Based Weight [15]: Compared with SR, which searches a global SR, local SR (LSR) performs sparse decomposition in local neighborhood. For each data point $\mathbf{x}_i$, LSR first finds its k-nearest neighbors as dictionary and then decomposes $\mathbf{x}_i$ as a sparse linear combination of the k points using the same rule as SR. It is worth noting that both LSR and LLR minimize the reconstruction error in local neighborhood but use different constraints for the reconstruction coefficients.

III. LML-Based Graph Construction

A. Relating LML and Graph-Based SSL via a Graph Representation

Graph-based SSL and LML have a close relationship: The former utilizes the data manifold in the graph Laplacian regularization term, which constrains the predictions to be smooth with respect to the data manifold; the latter that is proposed for nonlinear dimensionality reduction attempts to recover the real data manifold by preserving the local geometries of the overlapping neighborhoods. Moreover, in SSL, the data manifold is described by a graph, which plays the most important role for prediction. Since each LML method can be demonstrated to correspond to a specific graph [17], which summarizes all the relationships between the data points and is able to characterize certain geometric properties of the data, we employ the LML for graph construction in SSL.

According to the graph embedding framework [17], if an LML method can be reformulated in the following graph embedding framework:

$$\mathbf{Z}^* = \arg\min_{\mathbf{Z}\mathbf{Z}^{\mathrm{T}}=\mathbf{I}} \sum_{i,j=1}^{N} \mathbf{W}_{ij} \|\mathbf{Z}_i - \mathbf{Z}_j\|^2 = \arg\min_{\mathbf{Z}\mathbf{Z}^{\mathrm{T}}=\mathbf{I}} \mathrm{Tr}(\mathbf{Z}\mathbf{L}\mathbf{Z}^{\mathrm{T}}) \tag{9}$$

the graph structure of the LML method can be obtained. Therefore, the way to obtain the LML-based graph is to formulate the LML method as the optimization problem in (9). Here, $\mathbf{W}_{ij}$ denotes the edge weight between two points $\mathbf{x}_i$ and $\mathbf{x}_j$, $\mathbf{L}$ is the graph Laplacian matrix, $\mathbf{Z} \in \mathrm{R}^{d\times N}$ represents the embedding coordinates of the original high-dimensional data and is obtained from the eigenvectors corresponding to the smallest $2 \sim (d+1)$ eigenvalues of $\mathbf{L}$ (the eigenvector that corresponds to the smallest zero eigenvalue is a uniform vector with equal elements and is discarded). A common step of the three LML methods (LLE, LTSA, and LE) is that they eventually solve a sparse eigenvalue problem, and the embedding results are the eigenvectors corresponding to the smallest $2 \sim (d+1)$ eigenvalues of a matrix, which can be demonstrated to be a graph Laplacian matrix. Therefore, it is natural to formulate the three LML methods via a graph embedding representation.

It should be noted that the graph is an outcome of the LML method. The graph associated with an LML method is fixed, and therefore, we did not design the graph for the LML method but attempt to derive and analyze the graph structure that is produced by the LML method. In the following, we formulate the three LML methods (LLE, LTSA, and LE) in graph embedding formula (9), provide the $\mathbf{W}$ definitions, and discuss the properties of the LML-based regularization term.

B. LML-Based Graph

A common step of the three commonly used LML methods (LLE, LTSA, and LE) is that they are initiated by constructing an adjacency graph G_0 by using the kNN approach, where each data point is connected to its k-nearest neighbors.

1) LLE-Based Graph [15]: In LLE, the local properties of each neighborhood are represented by the linear coefficients that best reconstruct each data point from its neighbors. Define an $N \times N$ matrix $\mathbf{S}$ with S_{ij} denoting the reconstruction coefficient of $\mathbf{x}_i$ from its jth neighbor $\mathbf{x}_j$ and is calculated by

$$S_{ij} = \arg\min \left\| \mathbf{x}_i - \sum_{\mathbf{x}_j \in N(\mathbf{x}_i)} S_{ij}\mathbf{x}_j \right\|^2 \quad \text{s.t.} \sum_{\mathbf{x}_j \in N(\mathbf{x}_i)} S_{ij} = 1. \tag{10}$$

The embedding is obtained by retaining the reconstruction weights in low-dimensional space, which is constructed from the smallest $2 \sim (d+1)$ eigenvectors of the matrix $\mathbf{L}_{(\mathrm{lle})} = (\mathbf{I}-\mathbf{S})^{\mathrm{T}}(\mathbf{I}-\mathbf{S})$ [29]. Therefore, the LLE can be formulated as the graph embedding problem

$$\mathbf{Z}^* = \arg\min_{\mathbf{Z}\mathbf{Z}^{\mathrm{T}}=\mathbf{I}} \mathrm{Tr}\left(\mathbf{Z}\mathbf{L}_{(\mathrm{lle})}\mathbf{Z}^{\mathrm{T}}\right). \tag{11}$$

Since this matrix $\mathbf{L}_{(\mathrm{lle})}$ can be considered as a graph Laplacian matrix (refer to [17] for demonstration), we can define the adjacency matrix as $\mathbf{W}_{(\mathrm{lle})} = \mathbf{D} - \mathbf{L}_{(\mathrm{lle})} = \mathbf{S} + \mathbf{S}^{\mathrm{T}} - \mathbf{S}^{\mathrm{T}}\mathbf{S}$, with degree matrix $\mathbf{D} = \mathbf{I}$. The graph edge weight between $\mathbf{x}_i$ and $\mathbf{x}_j$ is given by [17]

$$\mathbf{W}_{(\mathrm{lle})_{ij}} = \begin{cases} S_{ij} + S_{ji} - \sum_r S_{ri}S_{rj}, & i \neq j \\ 0, & i = j. \end{cases} \tag{12}$$

The edge weight $\mathbf{W}_{(\mathrm{lle})ij}$ is determined by all the neighborhoods that include $\mathbf{x}_i$ and $\mathbf{x}_j$. Moreover, if $\mathbf{x}_i \notin N(\mathbf{x}_j)$ and $\mathbf{x}_j \notin N(\mathbf{x}_i)$, but both $\mathbf{x}_i$ and $\mathbf{x}_j$ are contained in the neighborhood of another point, there will be a weighted edge between the two points. Therefore, compared with the adjacency structure G_0, the LLE-based graph G includes more connections. As long as $\mathbf{x}_i$ and $\mathbf{x}_j$ are included in some neighborhood, there will be an edge between the two points in G.

Moreover, we can analyze the LLE-based Laplacian regularizer from another perspective. With the definition of graph weight matrix $\mathbf{W}_{(\mathrm{lle})}$, the LLE-based Laplacian regularizer can be transformed as in [34], [35]

$$\sum_{i,j=1}^{N} \mathbf{W}_{(\mathrm{lle})ij} \|\mathbf{f}_i - \mathbf{f}_j\|^2 = \sum_{i=1}^{N} \left\| \mathbf{f}_i - \sum_{\mathbf{x}_j \in N(\mathbf{x}_i)} S_{ij}\mathbf{f}_j \right\|^2 \tag{13}$$

where the right side of the equation is called the LLE regularizer (LLE-Reg) in [35]. Compared with the Laplacian regularizer on the left side that constrains the smoothness of predictions on graph, the LLE-Reg constrains that the predictions preserve the local geometry that is measured by the reconstruction weights

in each neighborhood. Therefore, the GFHF_LLE yields the predictions that preserve the local properties subject to $\mathbf{F}_l = \mathbf{Y}_l$.

2) LTSA-Based Graph: In LTSA, local geometry is described by the local tangent space of each data point. Let $\mathbf{X}_r$ be the k neighbors including $\mathbf{x}_r$ itself, and $\mathbf{\Theta}_r \in \mathrm{R}^{d\times k}$ of dimensionality d be the local tangent coordinates of $\mathbf{X}_r$ obtained by principal component analysis. The embedding is obtained by aligning all the overlapping local structures via the eigendecomposition of alignment matrix $\mathbf{L}_{(\mathrm{ltsa})}$ with $\mathbf{L}_{(\mathrm{ltsa})}(I_r, I_r) \leftarrow \mathbf{L}_{(\mathrm{ltsa})}(I_r, I_r) + \mathbf{U}_r$, where I_r are the indexes of $\mathbf{X}_r$, $\mathbf{U}_r = \mathbf{H}(\mathbf{I}_k - \mathbf{\Theta}_r^{\mathrm{T}}(\mathbf{\Theta}_r\mathbf{\Theta}_r^{\mathrm{T}})^{-1}\mathbf{\Theta}_r)$, $\mathbf{H} = \mathbf{I}_k - \mathbf{e}\mathbf{e}^{\mathrm{T}}/k$ is a $k \times k$ centering matrix, and $\mathbf{e}$ is a $k \times 1$ uniform vector. The eigenvectors corresponding to the smallest $2 \sim (d+1)$ eigenvalues of $\mathbf{L}_{(\mathrm{ltsa})}$ are used as embedding results. Therefore, it is natural for LTSA to be formulated as the graph embedding problem

$$\mathbf{Z}^* = \arg\min_{\mathbf{Z}\mathbf{Z}^{\mathrm{T}}=\mathbf{I}} \mathrm{Tr}\left(\mathbf{Z}\mathbf{L}_{(\mathrm{ltsa})}\mathbf{Z}^{\mathrm{T}}\right). \tag{14}$$

If we define the adjacency matrix $\mathbf{W}_{(\mathrm{ltsa})} = \mathbf{D} - \mathbf{L}_{(\mathrm{ltsa})}$, with degree matrix $\mathbf{D} = \mathbf{I}$, we obtain the graph edge weight of LTSA as

$$\mathbf{W}_{(\mathrm{ltsa})ij} = \begin{cases} \sum\limits_{(\mathbf{x}_i,\mathbf{x}_j)\in N(\mathbf{x}_r)} \left(\frac{1}{k} + \frac{1}{k-1}\boldsymbol{\theta}_{ri}^{\mathrm{T}}\mathbf{\Lambda}_r^{-1}\boldsymbol{\theta}_{rj}\right), & i \neq j \\ 0, & i = j \end{cases} \tag{15}$$

where $\boldsymbol{\theta}_{ri}$ and $\boldsymbol{\theta}_{rj}$ are the local tangent coordinates of $\mathbf{x}_i$ and $\mathbf{x}_j$ in $N(\mathbf{x}_r)$, and $\mathbf{\Lambda}_r$ is composed of the leading d eigenvalues of the covariance matrix of $\mathbf{X}_r$. The derivation of (15) can be found in Appendix A.

The edge weight $\mathbf{W}_{(\mathrm{ltsa})ij}$ is determined by all the neighborhoods that includes $\mathbf{x}_i$ and $\mathbf{x}_j$, where the term $\boldsymbol{\theta}_{ri}^{\mathrm{T}}\mathbf{\Lambda}_r^{-1}\boldsymbol{\theta}_{rj}$ indicates the inner product of whitened data $(\mathbf{\Lambda}_r^{-1/2}\boldsymbol{\theta}_{ri})$ and $(\mathbf{\Lambda}_r^{-1/2}\boldsymbol{\theta}_{rj})$ in the local tangent space or the Mahalanobis distances between $\mathbf{x}_i$ and $\mathbf{x}_j$ in the original space. Therefore, the edge weights are able to characterize the geometric properties of the related neighborhoods.

Moreover, the LTSA-based Laplacian regularizer satisfies the following equation:

$$\sum_{i,j=1}^{N} \mathbf{W}_{(\mathrm{ltsa})ij}\|\mathbf{f}_i - \mathbf{f}_j\|^2 = \sum_{i=1}^{N} \|\mathbf{F}_i\mathbf{H} - \mathbf{T}_i\mathbf{\Theta}_i\|^2 \tag{16}$$

where $\mathbf{F}_i \in \mathrm{R}^{C\times k}$ denotes the predictions of a local neighborhood of $\mathbf{x}_i$, which contains k data points including $\mathbf{x}_i$ itself, $\mathbf{T}_i$ is a local affine transformation matrix, and the term $\|\mathbf{F}_i\mathbf{H} - \mathbf{T}_i\mathbf{\Theta}_i\|$ indicates the local reconstruction error. We refer to the right side of the equation as LTSA-Reg. Via the minimization of the LTSA-Reg, the predictions $\mathbf{F}_i$ are constrained to preserve as much of the local geometry measured by the local tangent coordinates $\mathbf{\Theta}_i$ as possible. Therefore, the GFHF_LTSA is able to yield prediction results that preserve the local properties subject to $\mathbf{F}_l = \mathbf{Y}_l$. Equation (16) is derived in Appendix B.

3) LE-Based Graph: In LE, the local property is obtained by pairwise distances between neighbors, where the distance is normally calculated by an HK function with parameter σ. LE naturally follows the graph embedding formulation. The adjacency matrix $\mathbf{W}_{(\mathrm{le})}$ is calculated using (6), which is equal to the HK-based weight, and the normalized graph Laplacian matrix is obtained by $\mathbf{L}_{(\mathrm{le})} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}_{(\mathrm{le})}\mathbf{D}^{-1/2}$. It is easy to demonstrate that GFHF_LE is equivalent to GFHF_HK.

C. Properties of LML-Based Graphs

We summarize the properties of LLE- and LTSA-based graphs as follows.

1) LLE- and LTSA-based graph Laplacian regularizers constrain the prediction vectors to preserve the local geometry of each neighborhood. This implies that the SSL with LML-based graph can retain the desirable properties of the LML method, which is beneficial for classification, since the resulting predictions can reveal the nonlinear relationships of the high-dimensional data.
2) Each LML method corresponds to a specific graph structure. For LLE and LTSA, we first seek the Laplacian matrix, from which the edge weights can then be derived. Therefore, the graphs of LLE and LTSA are obtained from the Laplacian matrix. It is different from the HK(LE)-, LLR-, SR-, and LSR-based graph, where the Laplacian matrix is calculated from the edge weights.
3) The graph is an outcome of the LML method. Compared with the graph G_0 constructed by the kNN method, we found that the derived LLE- and LTSA-based graphs have more connections. In graph G_0, two points $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected if they are neighbors to each other ($\mathbf{x}_i \in N(\mathbf{x}_j)$ or $\mathbf{x}_j \in N(\mathbf{x}_i)$); in LLE- and LTSA-based graphs, $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected as long as they are included in some neighborhood ($\mathbf{x}_i \in N(\mathbf{x}_j)$ or $\mathbf{x}_j \in N(\mathbf{x}_i)$ or $\{\mathbf{x}_i, \mathbf{x}_j\} \in N(\mathbf{x}_k)$). In the case of $\{\mathbf{x}_i, \mathbf{x}_j\} \in N(\mathbf{x}_k)$, $\mathbf{x}_i$ and $\mathbf{x}_j$ are likely to be related, and LLE and LTSA can characterize the relations between these points, which may produce graphs with more useful information and thus improve the classification. Moreover, the edge weight between the two points is determined by all the neighborhoods that include them, whereas HK, LLR, and LSR calculate weights only in one neighborhood.
4) The HK-, LLR-, and SR-, and LSR-based graph weights are nonnegative, whereas LLE- and LTSA-based graph weights may include negative weights. The negative weight between two points denotes the dissimilarity and can decrease the possibility that they belong to the same class. Since the positive and negative weights together characterize the local geometry, these negative weights have a meaning and value for measuring the local geometry and, thus, are important for prediction. It is worth noting that for semisupervised classification using iterative label propagation, the negative weights are not suitable for propagating labels since the convergence cannot be achieved.
5) Computational complexity of obtaining LML-based graphs is based on selection of nearest neighbors, which requires $O(DN^2)$ and calculation of the Laplacian matrix (LLE is $O(DNk^3)$, and LTSA is $O(DNk^3) + O(dk^2)$), where D is the dimensionality of the high-dimensional data. The last step of GFHF is to predict the results of unlabeled data using (4), which is $O(u^3)$ since the inverse of the $u \times u\mathbf{L}_{uu}$ matrix is needed. Therefore, the overall complexity of GFHF_LLE and GFHF_LTSA is $O(DN^2) + O(DNk^3) + O(u^3)$ and $O(DN^2) + O(DNk^3) + O(dk^2) + O(u^3)$, respectively.

D. Related Works and Discussion

The following discussions outline the relationship between the proposed method and several well-known related works.

1) Graph Embedding Framework [17]: Yan unified dimensionality reduction methods within a graph embedding framework, where each method can be considered as the direct graph embedding of a specific intrinsic graph, and derived the graph weight definitions for many popular dimensionality reduction methods. Yan's work is the basis of our proposed method. However, the topic of their work is dimensionality reduction and has no relation to SSL.

2) SSML [36]: Yang proposed semisupervised ML (SSML) methods by considering prior information on the exact mapping of certain data points. The objective function is the same as (2), where $\mathbf{F}$ relates to the dimensionality reduction results, and $\mathbf{Y}_l$ denotes the prior embedding information of some data points. GFHF and SSML have the same format of objective function and solutions; however, the former is for classification, and the latter is for dimensionality reduction.

3) Unified View of Graph-Based SSL by Agovic and Banerjee [37]: Agovic and Banerjee proposed to unify label propagation, graph cuts, and SSML [36] by using a generalized label propagation formulation. The differences between their work and our method include the following: 1) The authors [37] related SSML and label propagation by specification of the prior information in SSML methods, but the intrinsic relations between the two methods are not clear. In our proposed method, we build relations between unsupervised LML and SSL through the *graph* and thus bring a family of LML-based graphs to SSL. 2) The authors [37] referred to the matrix $\mathbf{L}$ in SSML as "a suitable positive semidefinite matrix" and used $\mathbf{L}$ in the label propagation method, whereas in our study, the matrix $\mathbf{L}$ in LML methods is demonstrated to be the graph Laplacian matrix that is associated with a specific graph, and therefore, $\mathbf{L}$ can be applied to graph-based SSL methods. Hence, the matrix $\mathbf{L}$ has clearer meaning in our paper. 3) The edge weight formulations can be used for the out-of-sample extension, whereas it is difficult for [37] to process new data without the formulations of graph weight of ML methods.

IV. Out-of-Sample Extension

Graph-based SSL performs transductive classification on unlabeled data points. For classification of a new testing data point $\mathbf{x}_0$ outside the labeled and unlabeled data $\mathbf{X} = [\mathbf{X}_l, \mathbf{X}_u]$, a strategy is to fix the graph on $\mathbf{X}$ and predict the label of $\mathbf{x}_0$ from the labels of its neighbors in $\mathbf{X}$ [38].The Laplacian regularizer constrains the predictions to be smooth with respect to the graph, and therefore, the same smoothness criterion is also applied to the new testing point $\mathbf{x}_0$ [38], [12]

$$\frac{1}{2}\sum_{\mathbf{x}_i\in\mathbf{X},\mathbf{x}_i\in N(\mathbf{x}_0)} \mathbf{W}_{0i}\|\mathbf{f}_0-\mathbf{f}_i\|^2 \tag{17}$$

where $\mathbf{W}_{0i}$ denotes the weight between $\mathbf{x}_0$ and its neighbor $\mathbf{x}_i$, and $\mathbf{f}_i$ is the predicted result from transductive classification. Minimizing this term leads to the prediction of $\mathbf{x}_0$

$$\mathbf{f}_0 = \sum_{\mathbf{x}_i\in\mathbf{X},\mathbf{x}_i\in N(\mathbf{x}_0)} \mathbf{W}_{0i}\mathbf{f}_i. \tag{18}$$

For LLE-based weights in (12), $\mathbf{W}_{(\mathrm{lle})ij}$ is determined by all the neighborhoods that include $\mathbf{x}_i$ and $\mathbf{x}_j$. For the new testing point $\mathbf{x}_0$, we select its k-nearest neighbors from $\mathbf{X}$ (labeled and unlabeled data), and therefore, there is only one neighborhood $N(\mathbf{x}_0)$ that includes $\mathbf{x}_0$. As a result, the edge weight formula in (12) for $\mathbf{x}_0$ and its neighbor $\mathbf{x}_i$ can be simplified as $\mathbf{W}_{(\mathrm{lle})_{0i}} = S_{0i}$, where S_{0i} denotes the reconstruction coefficient of $\mathbf{x}_0$ from its neighbor $\mathbf{x}_i$.

For LTSA-based weights in (15), $\mathbf{W}_{(\mathrm{ltsa})ij}$ is also determined by all the neighborhoods that include $\mathbf{x}_i$ and $\mathbf{x}_j$. Since the new testing point $\mathbf{x}_0$ and its neighbor $\mathbf{x}_i$ are only included in one neighborhood $N(\mathbf{x}_0)$, (15) becomes $\mathbf{W}_{(\mathrm{ltsa})0i} = (1/k) + (1/(k-1))\boldsymbol{\theta}_0^{\mathrm{T}}\boldsymbol{\Lambda}_0^{-1}\boldsymbol{\theta}_i$, where $\boldsymbol{\theta}_0$ and $\boldsymbol{\theta}_i$ are the tangent coordinates of $\mathbf{x}_0$ and $\mathbf{x}_i$ in $N(\mathbf{x}_0)$, $\boldsymbol{\Lambda}_0$ is composed of the leading d eigenvalues of the covariance matrix of data from neighborhood $N(\mathbf{x}_0)$.

V. Experimental Results and Analysis

A. Data Description

Three hyperspectral images from two hyperspectral sensors were used for experiments. One was collected by the Hyperion instrument on the NASA EO-1 satellite, and the other two by the NASA Airborne Visible/Infrared Imaging Spectrometer (AVIRIS). Hyperion acquires 242-band data at 30-m spatial resolution, covering the 357–2576-nm portion of the spectrum in 10-nm bands. Uncalibrated and noisy bands were removed, and 145 bands remained. The Hyperion images utilized in the experiments were acquired over the Okavango Delta, Botswana (BOT) in May 2001. The 224-band AVIRIS data have 10-nm spectral resolution over the range of 400–2500 nm. The two images were collected over Kennedy Space Center (KSC) in March 1996 and Indiana Indian Pine (IND PINE) in 1992. The spatial resolutions of the two images are 18 and 20 m, respectively; the available bands for analysis of the two images are 176 and 220 after removal of noisy and water absorption bands. Fig. 1 shows the RGB images and ground reference information with class legends for the three scenes.

Six data sets were used for the experiments. In the BOT data, labeled data consist of nine identified land cover types in seasonal swamps, occasional swamps, and woodlands. Experiments include both overall classification results and accuracies from "difficult" classes with spectrally overlapping signatures. The first two data sets are from a class pair (class 3: Riparian; class 6: Woodlands), which are difficult to discriminate spectrally, and all the nine classes of the BOT images. In the KSC data, 13 land cover classes were labeled. The classes of Cabbage Palm Hammock (class 3) and Broad Leaf/Oak Hammock (class 6) are trees that grow in uplands; they have mixed spectral signatures with subtle differences and are difficult to classify. The two experimental data sets of KSC data are from classes 3 and 6 and classes 1–13. For 16-class IND PINE data, we selected data from nine classes (classes 2, 3, 5, 6, 8, 10, 11, 12, 14), which have a modest number of labeled data points for evaluation of the methods. In addition, data from class 3 (Corn, min) and class 12 (Soybeans, clean) are also used for experiments since they are difficult to discriminate during the early part of the growing season. Table I lists the class names

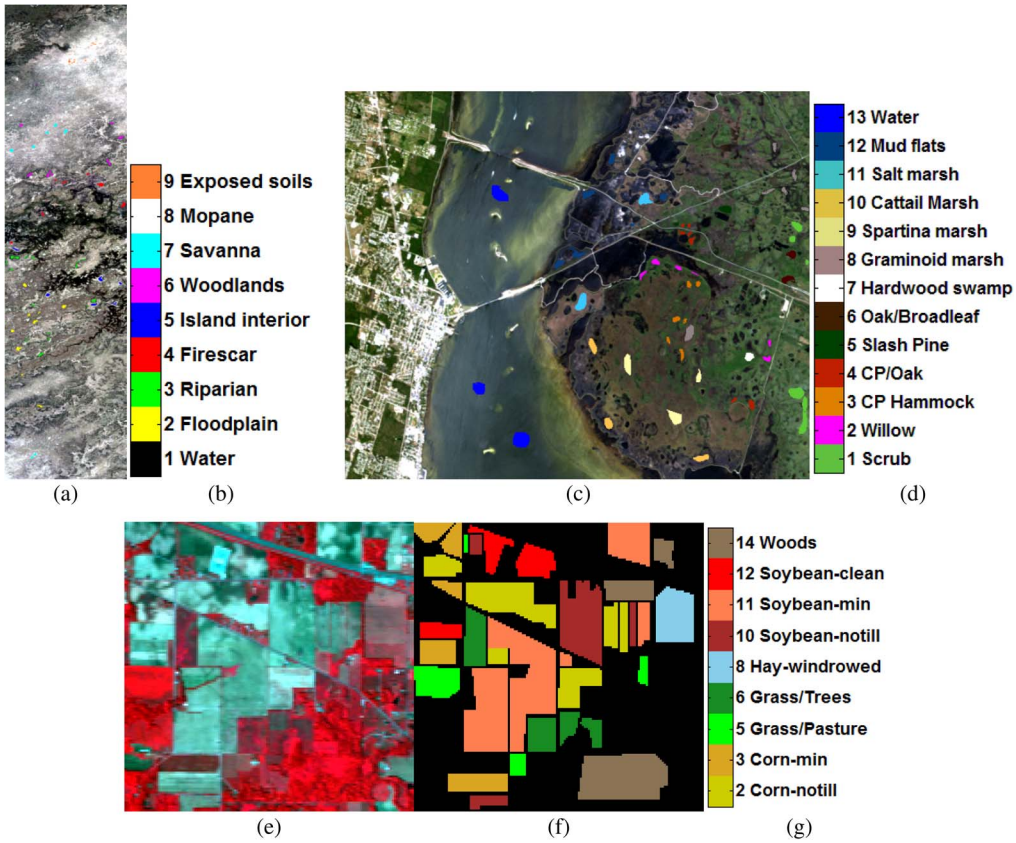


Fig. 1. RGB images and ground references. (a) BOT scene with ground reference (band 29, 23, 16 for red, green, and blue, respectively). (b) Class legend of BOT image. (c) KSC scene with ground reference (band 28, 19, 10 for red, green, and blue, respectively). (d) Class legend of KSC image. (e) IND PINE scene (band 57, 27, 17 for red, green, and blue, respectively). (f) Ground reference of IND PINE image. (g) Class legend of IND PINE image.

TABLE I
CLASS NAMES AND NUMBER OF SAMPLES

BOT		KSC		IND PINE	
ID	Class Name	ID	Class Name	ID	Class Name
1	Water (158)	1	Scrub (761)	2	Corn - No till (1434)
2	Floodplain (228)	2	Willow Swamp (243)	3	Corn – Min till (834)
3	Riparian (237)	3	Cabbage Palm Hammock (256)	5	Grass/pasture (497)
4	Firescar (178)	4	Cabbage Palm/Oak Hammock (252)	6	Grass/trees (747)
5	Island Interior (183)	5	Slash Pine (161)	8	Hay- windrowed (489)
6	Woodlands (199)	6	Broad Leaf/Oak Hammock (229)	10	Soybean – No till (968)
7	Savanna (162)	7	Hardwood Swamp (105)	11	Soybean – Min till (2468)
8	Mopane (124)	8	Graminoid Marsh (431)	12	Soybean – Clean (614)
9	Exposed Soils (111)	9	Spartina Marsh (520)	14	Woods (1294)
		10	Cattail Marsh (404)		
		11	Salt Marsh (419)		
		12	Mud Flats (503)		
		13	Water (927)		

and number of samples in each class of the BOT, KSC, and IND PINE data.

B. Results of Classification Experiments

The GFHF classifier with six different graphs was applied to the six data sets; the six graph construction methods are HK (LE), LLR, SR, LSR, LLE, and LTSA, where HK (LE) denotes that HK and LE produce the same graph. Except for SR that obtains the adjacency graph and edge weights simultaneously, the other five graphs utilize kNN for adjacency graph construction with spectral angle as the similarity measurement to search neighbors, because it favors spectral shape over spectral magnitude, which is beneficial to hyperspectral data. In addition, we conducted supervised classification using support vector machine (SVM) with an RBF kernel and kNN with $k = 1$ for comparison. We randomly divided each data set into three sets: labeled data, unlabeled data, and testing data, where the number of labeled data points ranged from 5 to 50 per class, and 70% of the remainder as unlabeled data for transduction and 30% as new testing data for induction. Twenty replications of the experiments were performed, and the average accuracies were calculated for evaluation. For parameters, except for the

TABLE II
CLASSIFIER PARAMETERS SELECTED BY LOO ERROR

GFHF #Label=30 per class	BOT		KSC		IND PINE	
	C3,6	C1-9	C3, 6	C 1-13	C 3,12	C2,3,5,6, 8,10,11, 12,14
HK(LE) (k[5,13],σ[0.1,10])	k=7; σ=0.5	k=7; σ=0.1	k=13; σ=0.1	k=7; σ=0.5	k=7; σ=1	k=7; σ=0.1
LLR (k[5,50])	k=50	k=50	k=20	k=35	k=50	k=20
LSR (k[10,50])	k=50	k=50	k=50	k=50	k=50	k=20
LLE (k[5,50])	k=50	k=50	k=50	k=50	k=50	k=50
LTSA (k[15,50],d[10,40])	k=50; d=25	k=40; d=20	k=50; d=25	k=50; d=25	k=50; d=40	k=50; d=40

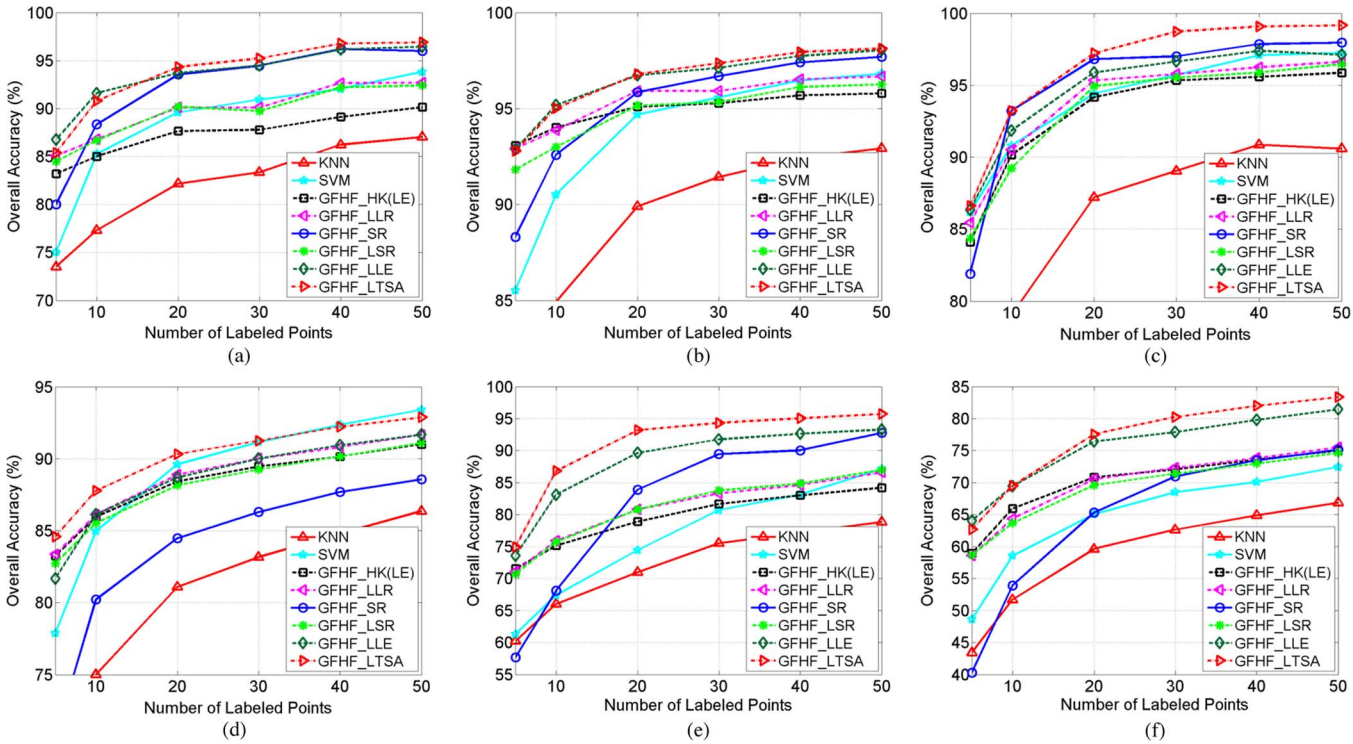


Fig. 2. Comparison of different classifiers (GFHF with six graphs, SVM, and kNN). (a) BOT classes 3 and 6. (b) BOT classes 1–9. (c) KSC classes 3 and 6. (d) KSC classes 1–13. (e) IND PINE classes 3 and 12. (f) IND PINE classes 2, 3, 5, 6, 8, 10, 11, 12, and 14.

k used in kNN adjacency graph construction, HK (LE) has the parameter σ in the HK function, and LTSA has the parameter d indicating the dimensionality of local tangent space. Parameter tuning was conducted by leave-one-out (LOO) classification error [35]. The range of parameter values used in LOO and the optimal parameter combinations with the number of labeled data points being equal to 30 per class of the six data sets are listed in Table II.

The classification results of the eight algorithms with best parameters on the six data sets are shown in Fig. 2, where the x-axis indicates the number of labeled points, and the y-axis denotes the overall accuracies (OAs). Several observations can be obtained: 1) LTSA-based graph performed the best for all the data sets with different numbers of labeled points; LLE also obtained higher accuracies than the other four graph construction methods for most of the data sets, which indicates that the corresponding graphs have good ability to describe the data manifold. 2) GFHF_SR performed better on data sets from two classes than for multiple classes. For large-scale data sets (KSC 13 classes and IND PINE 9 classes), SR obtained the lowest accuracies, which may be due to the large dictionary. Moreover, SR cannot obtain satisfactory results when the number of labeled points is very low. 3) Performing the SR locally, GFHF_LSR outperformed GFHF_SR on KSC 13 classes and IND PINE 9 classes, but cannot obtain higher accuracies on the other four data sets when the number of labeled data points is larger than 10. Moreover, LSR and LLR generate similar results for most of the data sets. 4) LLR achieved superior performance than HK (LE). 5) GFHF_HK(LE), which utilizes the most common edge weighting method, had better performance than SVM when the number of labels was very low. However, as the number of labeled data points increase, SVM outperformed GFHF_HK(LE). 6) Generally, the advanced graph construction methods (LLR, SR, LSR, LLE, and LTSA) yielded better OA than the original HK (LE) method, since the latter only considers pairwise similarity between data points, whereas the former methods calculate the weights by exploiting information between neighbors, resulting in better discriminative abilities for classification. 7) When the number of labeled data is equal to or larger than 10, the LLE- and LTSA-based graphs produced significant improvements relative to HK (LE)-based graph for all the data sets. However, when the number of labeled data is very small, LLE- and LTSA-based graphs performed similar to HK (LE) for BOT C1–9 and KSC C1–13. It seems that the information included in the advanced graphs may not be well explored if the prior knowledge is very limited.

The performances of GFHF with six different graphs were also evaluated on a per-class basis, which is arguably a better indicator of performance than overall classification accuracy. Fig. 3 shows the classification results for multiple classes of BOT, KSC, and IND PINE data when the number of labeled data points is 30. It can be seen that LLE- and LTSA-based graphs produced superior performances for most of the classes, particularly for the classes that are difficult to discriminate, e.g., BOT classes 3 and 6; KSC classes 4, 5, and 6; and IND PINE classes 2, 3, 11, and 12. For the KSC data, LSR obtained higher OA than SR, but produced lower accuracies on the "difficult" classes (classes 4, 5, and 6). HK(LE), LLR, SR, and LSR resulted in similar OA on IND PINE data, but the performances on different classes varied; for example, SR outperformed the others on class 2, but yielded more misclassifications on classes 10 and 11.

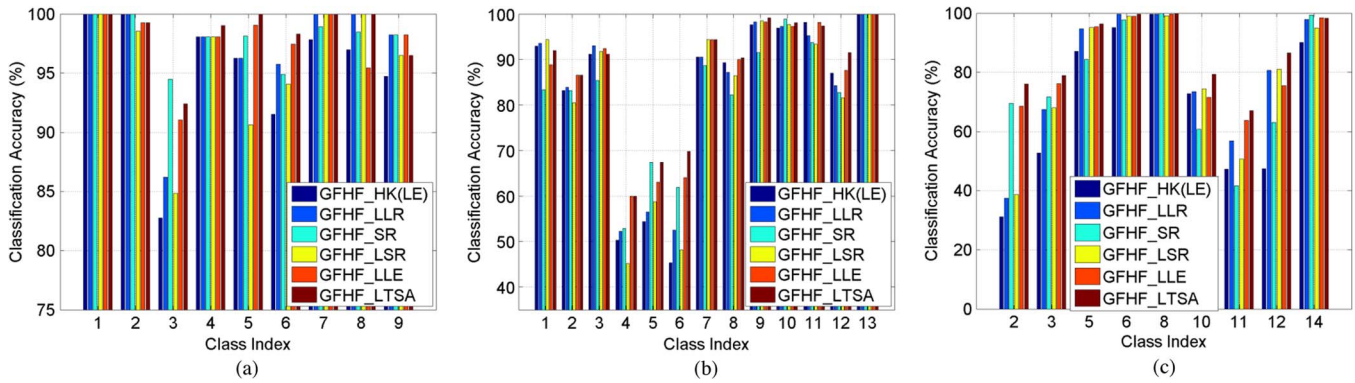


Fig. 3. Comparison of different graph construction methods on classification accuracy of each class. (a) BOT classes 1–9. (b) KSC classes 1–13. (c) IND PINE classes 2, 3, 5, 6, 8, 10, 11, 12, and 14.

TABLE III
COMPUTATIONAL TIME OF DIFFERENT GRAPHS (SECONDS)

KSC Data	GFHF_HK(LE)	GFHF_LLR	GFHF_SR	GFHF_LSR	GFHF_LLE	GFHF_LTSA
2 classes (357 points)	0.06	2.91	5.31	1.64	0.17	0.76
13 classes (3764 points)	5.97	101.92	262.77	20.32	4.96	36.83

Using a machine with Intel Xeon CPU E5-2620 and 16-G RAM, the computational time of different graph-based GFHF classifiers on two data sets (KSC C3, 6 and KSC C1–13) is shown in Table III. Experiments were conducted with parameters listed in Table II with 30 labeled data points. GFHF_HK(LE) and GFHF_LLE require least time. It is worth noting that LLE is very fast because in the MATLAB code, it does not actually compute the inverse of the Gram matrix in each neighborhood but uses the Gaussian elimination method in matrix division to calculate the reconstruction coefficients. GFHF_SR is the most time consuming since SR selects neighbors and calculates weights globally. GFHF_LSR that performs SR locally can effectively decrease the running time compared with GFHF_SR. GFHF_LLR is much slower than GFHF_LLE since the computation of fully constrained reconstruction coefficients is expensive particularly when the value of k is large. The computational time of GFHF_LTSA is acceptable.

C. Sensitivity Analysis of Parameters in LLE- and LTSA-Based Graph Construction

Data from BOT 9 classes, KSC 13 classes, and IND PINE 9 classes were used to conduct sensitivity analysis for parameters in LLE- and LTSA-based graph construction. LLE includes one free parameter k, the number of neighbors in the kNN-based adjacency graph; LTSA requires k and another parameter d, which denotes the dimensionality of the local tangent space. For LLE, Fig. 4(a)–(c) plotted the OA versus the number of labeled data with k as a parameter, where the value of k was tested from 5 to 50 with a step of 5. It can be seen that larger values of k performed better than small values, which suggests that LLE requires more data to model the geometry of hyperspectral data. Moreover, GFHF_LLE produced good results with $k > 20$, which suggests that it is robust relative to this parameter over a large range of values. For LTSA, Fig. 4(d)–(f) plotted the OA versus d with k as a parameter when the number of labeled points is 30, where the value of k was tested from 15 to 50 with a step of 5, and d was investigated from 10 to 40 with a step of 5 (k should be greater than d [30]). With fixed d, larger values of k obtained superior performances, but with fixed k, the best value of d varied for different data sets. It can be observed that $d = 20$ is suitable for most of the data sets except for IND PINE (9 classes), which required a large value of d. Therefore, LTSA is sensitive to parameters k and d. Generally, we can select a large value of k and a suitable value of d based on the LOO method.

D. Out-of-Sample Extension

For new testing data, the classification performances of GFHF with six graph construction methods are shown in Fig. 5, where the results for BOT classes 3, 6 and KSC classes 3, 6 are listed. It can be seen that the performances of the six graph construction methods on testing data were similar to their performances on unlabeled data sets in Fig. 2(a) and (c), which indicates the effectiveness of the out-of-sample extension method.

We also conducted induction of all data points in the whole image (IND PINE and BOT) using 50 labeled data per class. Since IND PINE image contains abundant referenced data point, we only illustrated the results on the referenced data shown in Fig. 6, where (a)–(f) provide the results of the GFHF classifier with six graph construction methods, and (g) shows the class legend. It can be seen that the results from GFHF_LLE and GFHF_LTSA outperformed the other four classifiers, particularly for classes 2, 3, 11, and 12. SR performed the worst; it had many misclassifications of classes 10 and 11. LE, LLR, and LSR yielded many false predictions since they cannot distinguish classes 2 and 11, 12 very well. The OA of HK(LE)-, LLR-, SR-, LSR-, LLE-, and LTSA-based classification were 75.49%, 77.55%, 74.08%, 77.03%, 83.57%, and 84.54%, respectively.

The BOT data (size of 1476 $\times$ 256) contain two major ecosystem components defined by the absence or presence of flooding: upland and wetland [39]. Classification results using GFHF with six graphs were similar in most areas. However, distinct differences can still be found in several local regions, where some misclassified pixels can be identified according to spectral analysis and domain knowledge. We selected two local regions to illustrate the different performances of the six graph construction methods. Fig. 7(a) shows a region of size 66 $\times$ 66

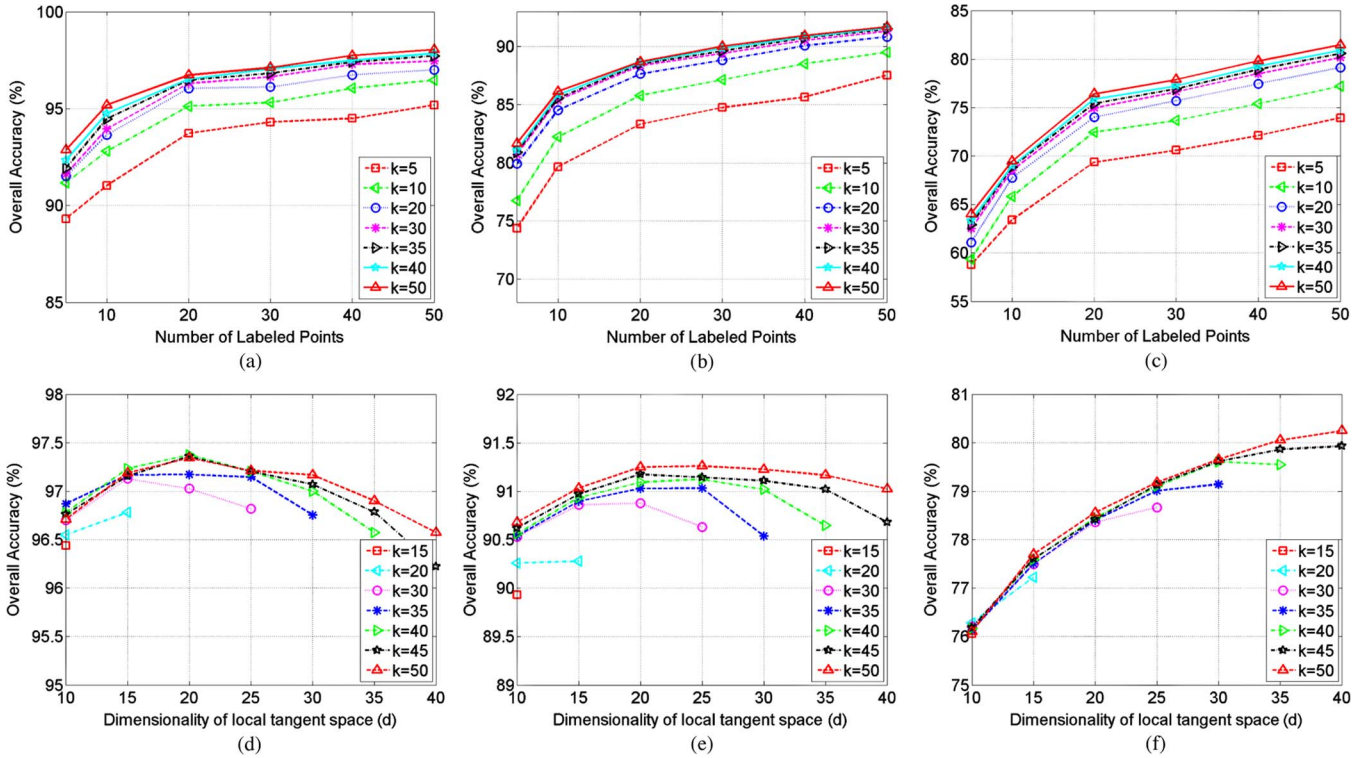


Fig. 4. Sensitivity analysis of parameters in LLE- and LTSA-based graph construction. (a) GFHF_LLE on BOT 9 classes. (b) GFHF_LLE on KSC 13 classes. (c) GFHF_LLE on IND PINE 9 classes. (d) GFHF_LTSA on BOT 9 classes. (e) GFHF_LTSA on KSC 13 classes. (f) GFHF_LTSA on IND PINE 9 classes.

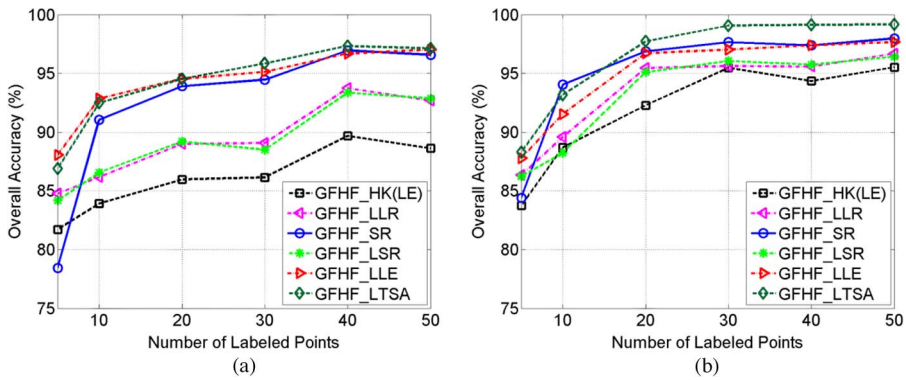


Fig. 5. Induction to new testing data. (a) BOT classes 3 and 6. (b) KSC classes 3 and 6.

on the upper right corner of the scene. This area is a wetland, containing water (black), floodplain (yellow), riparian (green), and firescar (red) around the water. Fig. 7(b)–(g) illustrates the classification results with the class legend shown in Fig. 7(h). It can be seen that woodland (purple) exists in all the results, which is determined as a misclassification, since it is a category of upland and should not exist in this wetland region. The misclassification occurs because the woodland (purple) and riparian trees (green), which are near the channels, have similar spectra. As shown, the GFHF_LTSA and GFHF_LLE performed best with the least misclassified pixels; GFHF_LLR and GFHFP_LSR generated fewer misclassifications than GFHF_LE; GFHF_SR produced a different result, where firescar (red) cover type was missed. Fig. 8(a) shows an upland region centered in (160, 79), which primarily contains land cover types of savanna (light blue) and exposed soil (orange). The island interior (dark blue) was identified as a misclassification. As shown in Fig. 8(b)–(g), the GFHF_LTSA and GFHF_LLE yielded satisfactory results with few classification errors, LE produced a lot of misclassifications, LLR and LSR cannot separate savanna and island interior well, and SR classified most of the pixels as exposed soil.

E. Discussion

Six graph edge weighting approaches (HK(LE), LLR, SR, LSR, LLE, and LTSA) were utilized and evaluated in this paper. The comparison and analysis are given as follows.

Generally, LLE and LTSA outperformed the other methods for most of the data sets, since LLE and LTSA are able to characterize the nonlinear relationships between data points, and the corresponding graphs have good ability to describe the data manifold. The LLE- and LTSA-based Laplacian regularizers constrain the prediction vectors to preserve the local

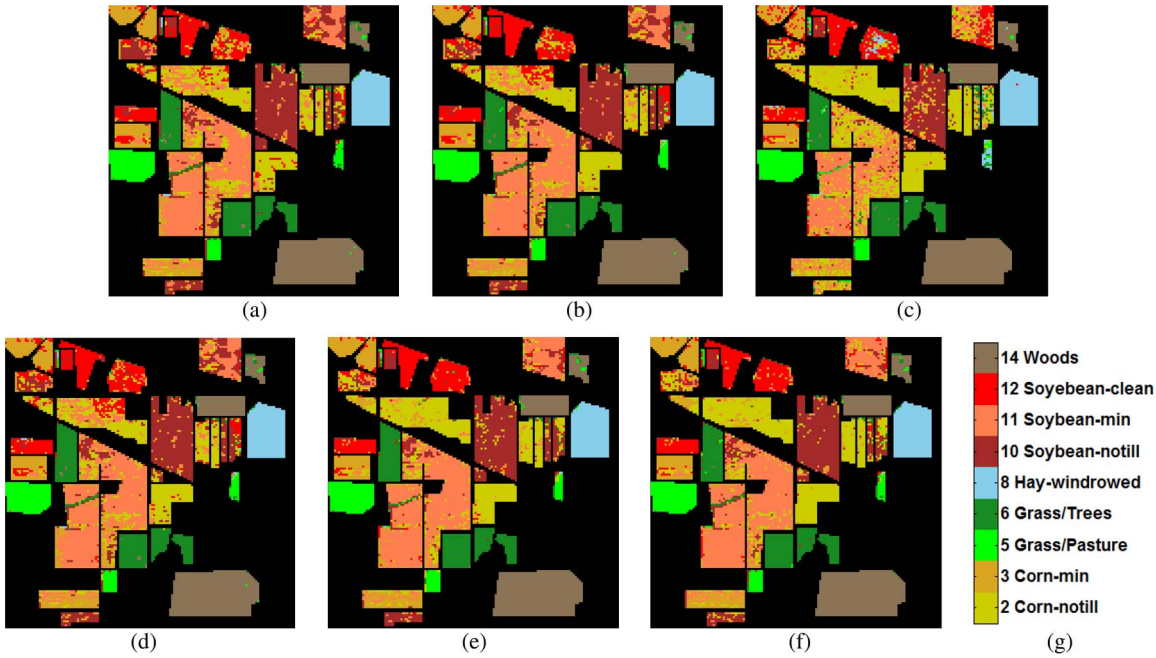


Fig. 6. Classification results of IND PINE referenced data. (a) GFHF_LE result. (b) GFHF_LLR result. (c) GFHF_SR result. (d) GFHF_LSR result. (e) GFHF_LLE result. (f) GFHF_LTSA result. (g) Class legend.

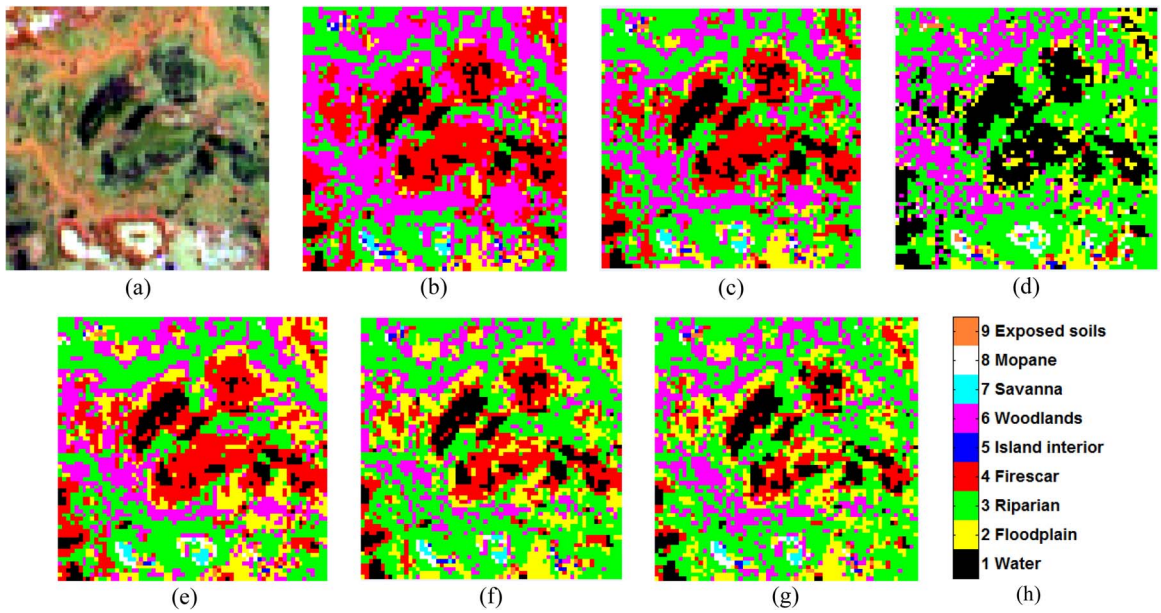


Fig. 7. Classification results of a local wetland region in BOT data. (a) Pseudo-RGB of the subscene. (b) GFHF_LE result. (c) GFHF_LLR result. (d) GFHF_SR result. (e) GFHF_LSR result. (f) GFHF_LLE result. (g) GFHF_LTSA result. (h) Class legend.

geometry of each neighborhood, which is beneficial for classification since the resulted predictions can reveal the nonlinear relationships of the high-dimensional data. Moreover, LLE and LTSA may contain negative weights that indicate the dissimilarities. Some of the nearest neighbors selected in the Euclidean space have similar spectra but may be from different classes. If some of the "false" neighbors have negative weights, the prediction power of the "false" classes will be decreased.

HK(LE) is the most simple and common method to calculate the edge weights. It is fast but often cannot obtain good performance, since it only considers the pairwise similarity between two data points, ignoring the neighborhood relationships and thereby potentially underutilizing available information. The LLR method considers neighborhood relationships and can obtain higher accuracies than HK(LE), but it requires the local reconstruction weights to be nonnegative and sum-to-one. As noted in [40], a possible disadvantage is: "this may degrade the reconstruction of data points that lie on the boundary of a manifold and outside the convex hull of their neighbors." SR did not perform well for large-scale data sets (KSC 13 classes and IND PINE 9 classes) in this study. Using a small dictionary for each data point, LSR achieves superior performance to SR for

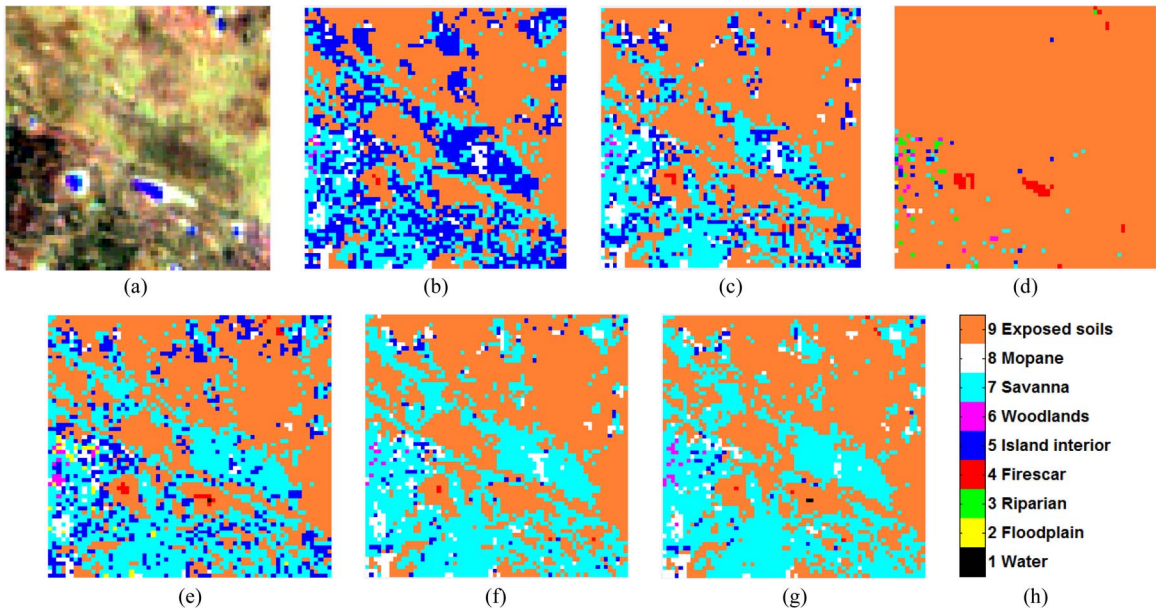


Fig. 8. Classification results of a local upland region in BOT data. (a) Pseudo-RGB of the subscene. (b) GFHF_LE result. (c) GFHF_LLR result. (d) GFHF_SR result. (e) GFHF_LSR result. (f) GFHF_LLE result. (g) GFHF_LTSA result. (h) Class legend.

large-scale data sets and requires much smaller computational time. However, LSR did not outperform SR for the other four data sets.

Considering both the classification performance and computational cost, LLE is the best choice for weighting the graph edge. Moreover, LLE is not sensitive to the value of the parameter k over a large range of values ($k > 20$). If we only pursue the highest classification accuracies, LTSA should be chosen. For data sets with small size, SR is also a good candidate to employ since it has no free parameters.

VI. Conclusion

This paper has developed the relation between LML and SSL via graphs, introducing a new family of graphs to SSL methods and extending the applications of LML methods. The SSL with LML-based graph can retain the good properties of the LML method, since the corresponding Laplacian regularizer constrains the prediction vectors to preserve the local geometry of each neighborhood. Experimental results demonstrated that the LLE- and LTSA-based graphs achieved higher accuracies than the other four popular graphs, which demonstrates that the graphs derived from LLE and LTSA have good ability to describe the data manifold. For the three LML methods (LLE, LTSA, and LE), LLE and LTSA are able to provide more effective graphs than LE, which is consistent with our previous investigations that LLE and LTSA resulted in superior performances to LE as dimensionality reduction preprocessing methods for hyperspectral image classification [21], [23]. In this paper, we only focused on LLE and LTSA and applied them to one SSL method (GFHF). However, other LML methods can also be incorporated into any graph-based SSL method. Moreover, these classifiers should be suitable for other hyperspectral images with high spatial resolution and other classification tasks.

It is worth noting that the graph-based SSL is based on a single-manifold structure. Considering multiple manifolds or hierarchical manifolds may improve the classification performance [41], [42]. Moreover, since the local curvature may vary in different neighborhoods, choosing adaptive neighborhoods may also increase the classification accuracies [43]. Our current work is in these directions.

Appendix A

LTSA algorithm can be reformulated as the graph embedding formulation in (9) with graph weight denoted as (15).

Proof: The alignment matrix of LTSA can be calculated by

$$\mathbf{L}_{(\mathrm{ltsa})}(I_r, I_r) \leftarrow \mathbf{L}_{(\mathrm{ltsa})}(I_r, I_r) + \mathbf{U}_r \tag{19}$$

where $\mathbf{U}_r \in \mathrm{R}^{k\times k}$ is calculated by

$$\begin{aligned}\mathbf{U}_r &= \mathbf{H}\left(\mathbf{I} - \boldsymbol{\Theta}_r^{\mathrm{T}}\left(\boldsymbol{\Theta}_r\boldsymbol{\Theta}_r^{\mathrm{T}}\right)^{-1}\boldsymbol{\Theta}_r\right)\\ &= \left(\mathbf{I} - \frac{1}{k}\mathbf{e}\mathbf{e}^{\mathrm{T}}\right)\left(\mathbf{I} - \boldsymbol{\Theta}_r^{\mathrm{T}}\boldsymbol{\Lambda}_r^{-1}\boldsymbol{\Theta}_r\right)\\ &= \mathbf{I} - \frac{1}{k}\mathbf{e}\mathbf{e}^{\mathrm{T}} - \boldsymbol{\Theta}_r^{\mathrm{T}}\boldsymbol{\Lambda}_r^{-1}\boldsymbol{\Theta}_r - \frac{1}{k}\mathbf{e}\mathbf{e}^{\mathrm{T}}\boldsymbol{\Theta}_r^{\mathrm{T}}\boldsymbol{\Lambda}_r^{-1}\boldsymbol{\Theta}_r. \end{aligned}\tag{20}$$

Since $\boldsymbol{\Theta}_r \in \mathrm{R}^{d\times k}$ is the local tangent coordinates obtained by principal component transformation, we have $\boldsymbol{\Theta}_r\mathbf{e} = \mathbf{0}$, and (20) becomes

$$\mathbf{U}_r = \mathbf{I} - \frac{1}{k}\mathbf{e}\mathbf{e}^{\mathrm{T}} - \boldsymbol{\Theta}_r^{\mathrm{T}}\boldsymbol{\Lambda}_r^{-1}\boldsymbol{\Theta}_r. \tag{21}$$

Further, we have $\mathbf{U}_r\mathbf{e} = \mathbf{e} - \mathbf{e} - \boldsymbol{\Theta}_r^{\mathrm{T}}\boldsymbol{\Lambda}_r^{-1}\boldsymbol{\Theta}_r\mathbf{e} = \mathbf{0}$, and thus, $\sum_j(\mathbf{U}_r)_{i,j} = 0, \quad i = 1, \ldots, k$, which means the summation of each row of $\mathbf{U}_r$ is equal to zero. The alignment matrix $\mathbf{L}_{(\mathrm{ltsa})}$ is composed by $\mathbf{U}_r$, and therefore,

$$\sum_j \mathbf{L}_{(\mathrm{ltsa})_{i,j}} = 0, \qquad i = 1, \ldots, N. \tag{22}$$

The row summation of matrix $\mathbf{L}_{(\mathrm{ltsa})}$ is zero, and thus, it can be considered as the Laplacian matrix of a graph. Therefore, we set the edge weight matrix $\mathbf{W}_{(\mathrm{ltsa})} = \mathbf{I} - \mathbf{L}_{(\mathrm{ltsa})}$, and degree matrix $\mathbf{D} = \mathbf{I}$.

Graph Laplacian matrix $\mathbf{L}_{(\mathrm{ltsa})}$ in terms of $\mathbf{x}_i$ and $\mathbf{x}_j$ is expressed as

$$\begin{aligned}\mathbf{L}_{(\mathrm{ltsa})_{ij}} &= \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathbf{X}_r} (\mathbf{U}_r)_{(i,j)} \\ &= \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathbf{X}_r} \left(\mathbf{I} - \frac{1}{k}\mathbf{e}\mathbf{e}^{\mathrm{T}} - \boldsymbol{\Theta}_r^{\mathrm{T}}\boldsymbol{\Lambda}_r^{-1}\boldsymbol{\Theta}_r\right)_{(i,j)} \\ &= \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathbf{X}_r} \left(\delta_{i,j} - \frac{1}{k} - \frac{1}{k-1}\boldsymbol{\theta}_{ri}^{\mathrm{T}}\boldsymbol{\Lambda}_r^{-1}\boldsymbol{\theta}_{rj}\right). \quad (23)\end{aligned}$$

Therefore, the edge weight between $\mathbf{x}_i$ and $\mathbf{x}_j$ is defined as $\mathbf{W}_{(\mathrm{ltsa})_{ij}} = \delta_{i,j} - \mathbf{L}_{(\mathrm{ltsa})_{ij}} = \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in N(\mathbf{x}_r)}((1/k) + (1/(k-1))\boldsymbol{\theta}_{ri}^{\mathrm{T}}\boldsymbol{\Lambda}_r^{-1}\boldsymbol{\theta}_{rj})$ when $i \neq j$, 0 otherwise. We have the graph embedding formula for LTSA

$$\begin{aligned}\mathbf{Z}^* &= \arg\min_{\mathbf{Z}\mathbf{Z}^{\mathrm{T}}=\mathbf{I}} \mathrm{Tr}\left(\mathbf{Z}\mathbf{L}_{(\mathrm{ltsa})}\mathbf{Z}^{\mathrm{T}}\right) \\ &= \arg\min_{\mathbf{Z}\mathbf{Z}^{\mathrm{T}}=\mathbf{I}} \sum_{i,j=1}^{N} \mathbf{W}_{(\mathrm{ltsa})_{ij}}\|\mathbf{Z}_i - \mathbf{Z}_j\|^2. \quad (24)\end{aligned}$$

APPENDIX B

The LTSA-based graph Laplacian regularizer can be transformed to LTSA-Reg defined in (16).

Proof: Let $\mathbf{E}_i = \mathbf{F}_i\mathbf{H} - \mathbf{T}_i\boldsymbol{\Theta}_i$ denote the local reconstruction error matrix. By minimizing $\|\mathbf{E}_i\|^2$, the optimal local affine transformation matrix is obtained as $\mathbf{T}_i = \mathbf{F}_i\mathbf{H}\boldsymbol{\Theta}_i^{\mathrm{T}}(\boldsymbol{\Theta}_i\boldsymbol{\Theta}_i^{\mathrm{T}})^{-1}$. Therefore, $\mathbf{E}_i = \mathbf{F}_i\mathbf{H}(\mathbf{I} - \boldsymbol{\Theta}_i^{\mathrm{T}}(\boldsymbol{\Theta}_i\boldsymbol{\Theta}_i^{\mathrm{T}})^{-1}\boldsymbol{\Theta}_i)$. Let $\mathbf{U}_i = \mathbf{H}(\mathbf{I} - \boldsymbol{\Theta}_i^{\mathrm{T}}(\boldsymbol{\Theta}_i\boldsymbol{\Theta}_i^{\mathrm{T}})^{-1}\boldsymbol{\Theta}_i)$, we have $\mathbf{E}_i = \mathbf{F}_i\mathbf{U}_i$. The LTSA-Reg becomes

$$\begin{aligned}\sum_{i=1}^{N}\|\mathbf{F}_i\mathbf{H} - \mathbf{T}_i\boldsymbol{\Theta}_i\|^2 &= \sum_{i=1}^{N}\|\mathbf{F}_i\mathbf{U}_i\|^2 = \|\mathbf{F}\mathbf{A}\mathbf{U}\|^2 \\ &= \mathrm{Tr}(\mathbf{F}\mathbf{A}\mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{F}^{\mathrm{T}}) = \mathrm{Tr}\left(\mathbf{F}\mathbf{L}_{(\mathrm{ltsa})}\mathbf{F}^{\mathrm{T}}\right) \quad (25)\end{aligned}$$

where $\mathbf{L}_{(\mathrm{ltsa})} = \mathbf{A}\mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}$, $\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_N]$ is 0-1 selection matrix such that $\mathbf{F}\mathbf{A}_i = \mathbf{F}_i$, $\mathbf{U} = \mathrm{diag}(\mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_N)$.

The right side of (25) is the Laplacian regularizer

$$\sum_{i,j=1}^{N}\mathbf{W}_{(\mathrm{ltsa})ij}\|\mathbf{f}_i - \mathbf{f}_j\|^2 = \mathrm{Tr}\left(\mathbf{F}\mathbf{L}_{(\mathrm{ltsa})}\mathbf{F}^{\mathrm{T}}\right). \quad (26)$$

Therefore, we have (16) for the LTSA-based Laplacian regularizer.

REFERENCES

[1] G. Camps-Valls, T. V. B. Marsheva, and D. Zhou, "Semi-supervised graph-based hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 10, pp. 3044–3054, Oct. 2007.

[2] W. Kim and M. M. Crawford, "Adaptive classification for hyperspectral image data using manifold regularization kernel machines," *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 11, pp. 4110–4121, Nov. 2010.

[3] L. Gómez-Chova, G. Camps-Valls, J. Muñoz-Mari, and J. Calpe, "Semi-supervised image classification with Laplacian support vector machines," *IEEE Geosci. Remote Sens. Lett.*, vol. 5, no. 3, pp. 336–340, Jul. 2008.

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

[5] X. Zhu, J. Lafferty, and Z. Ghahramani, "Semi-supervised learning literature survey," Comput. Sci., Univ. Wisconsin-Madison, Madison, WI, USA, TR-1530, Jul. 2008.

[6] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using Gaussian fields and harmonic functions," in *Proc. 20th Int. Conf. Mach. Learn.*, Washington, DC, USA, 2003, pp. 912–919.

[7] X. Zhu and Z. Ghahramani, "Learning from labeled and unlabeled data with label propagation," School Comput. Sci., Carnegie Mellon Univ., Pittsburgh, PA, USA, CMU-CALD-02-107, Jun. 2002.

[8] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *J. Mach. Learn. Res.*, vol. 7, pp. 2399–2434, Nov. 2006.

[9] D. Zhou, O. Bousquet, T. Navin Lal, J. Weston, and B. Scholkopf, "Learning with local and global consistency," in *Proc. Adv. Neural Inf. Process. Syst.*, S. Thrun, L. Saul, and B. Scholkopf, Eds., 2004, vol. 16, pp. 321–328.

[10] T. Jebara, J. Wang, and S. Chang, "Graph construction and b-matching for semi-supervised learning," in *Proc. 26th Int. Conf. Mach. Learn.*, Montreal, QC, Canada, 2009, pp. 441–448.

[11] C. Li, X. Qi, and B. Xiao, "An evaluation on different graphs for semi-supervised learning," in *Lecture Notes in Computer Science*, Y. Zhang *et al.*, Ed., 2012, vol. 7202, pp. 58–65.

[12] F. Wang and C. Zhang, "Label propagation through linear neighborhoods," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 1, pp. 55–67, Jan. 2008.

[13] S. Yan and H. Wang, "Semi-supervised learning by sparse representation," in *Proc. SIAM Int. Conf. Data Mining*, Sparks, NV, USA, Apr. 2009, pp. 792–801.

[14] H. Cheng, Z. Liu, and J. Yang, "Sparsity induced similarity measure for label propagation," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Kyoto, Japan, Sep. 2009, pp. 317–324.

[15] C. Li, J. Guo, and H. Zhang, "Local sparse representation based classification," in *Proc. 20th Int. Conf. Pattern Recog.*, Istanbul, Turkey, Aug. 2010, pp. 649–652.

[16] Y. Gu and K. Feng, "L1-graph semisupervised learning for hyperspectral image classification," in *Proc. IEEE Int. Symp. Geosci. Remote Sens.*, Munich, Germany, Jul. 2012, pp. 1401–1404.

[17] S. C. Yan *et al.*, "Graph embedding and extensions: A general framework for dimensionality reduction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 1, pp. 40–51, Jan. 2007.

[18] C. M. Bachmann *et al.*, "Bathymetric retrieval from hyperspectral imagery using manifold coordinate representations," *IEEE Trans. Geosci. Remote Sens.*, vol. 47, no. 3, pp. 884–897, Mar. 2009.

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

[20] C. M. Bachmann, T. L. Ainsworth, and R. A. Fusina, "Improved manifold coordinate representations of large-scale hyperspectral scenes," *IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 10, pp. 2786–2803, Oct. 2006.

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

[22] L. Ma, M. M. Crawford, and J. W. Tian, "Generalized supervised local tangent space alignment for hyperspectral image classification," *Electron. Lett.*, vol. 46, no. 7, pp. 497–498, Apr. 2010.

[23] M. M. Crawford, L. Ma, and W. Kim, "Exploring nonlinear manifold learning for classification of hyperspectral data," in *Optical Remote Sensing: Advances in Signal Processing and Exploitation Techniques*, S. Prasad, Ed. London, U.K.: Springer-Verlag, 2011, pp. 207–234.

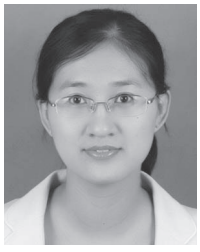
[24] D. Lunga, S. Prasad, M. M. Crawford, and K. Ersoy, "Manifold-learning-based feature extraction for classification of hyperspectral data," *IEEE Signal Process. Mag.*, vol. 31, no. 1, pp. 55–66, Jan. 2014.

[25] D. Lunga and O. Ersoy, "Spherical stochastic neighbor embedding of hyperspectral data," *IEEE Trans. Geosci. Remote Sens.*, vol. 51, no. 2, pp. 857–871, Feb. 2013.

[26] L. Ma, M. M. Crawford, and J. W. Tian, "Anomaly detection for hyperspectral images based on robust locally linear embedding," *J. Infrared, Millimeter, Terahertz Waves*, vol. 31, no. 6, pp. 753–762, Jun. 2010.

[27] L. Zhang, L. Zhang, D. Tao, and X. Huang, "Sparse transfer manifold embedding for hyperspectral target detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 2, pp. 1030–1043, Feb. 2014.

[28] J. B. Tenenbaum, V. Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, Dec. 2000.
[29] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, Dec. 2000.
[30] Z. Y. Zhang and H. Y. L. Zha, "Principal manifolds and nonlinear dimension reduction via local tangent space alignment," *SIAM J. Sci. Comput.*, vol. 26, no. 1, pp. 313–338, Dec. 2004.
[31] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Comput.*, vol. 15, no. 6, pp. 1373–1396, Mar. 2003.
[32] T. Han and D. G. Goodenough, "Investigation of nonlinearity in hyperspectral imagery using surrogate data methods," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 10, pp. 2840–2847, Oct. 2008.
[33] C. M. Bachmann and T. L. Ainsworth, "Exploiting manifold geometry in hyperspectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 3, pp. 441–454, Mar. 2005.
[34] J. Ham, D. D. Lee, and L. K. Saul, "Semisupervised alignment of manifold," in *Proc. Annu. Conf. Uncertainty Artif. Intell.*, Z. Ghahramani and R. Cowell, Eds., 2005, vol. 10, pp. 120–127.
[35] M. Wu and B. Scholkopf, "Transductive classification via local learning regularization," in *Proc. 11th Int. Conf. Artif. Intell. Statist.*, 2007, pp. 1529–1536.
[36] X. Yang, H. Fu, H. Zha, and J. Barlow, "Semi-supervised nonlinear dimensionality reduction," in *Proc. 23th Int. Conf. Mach. Learn.*, Pittsburgh, PA, USA, 2006, pp. 1065–1072.
[37] A. Agovic and A. Banerjee, "A unified view of graph-based semi-supervised learning: label propagation, graph-cuts, and embeddings," Dept. Comput. Sci. Eng., Univ. Minnesota, Minneapolis, MN, USA, TR-09-012, May 12, 2009.
[38] O. Delalleau, Y. Bengio, and N. L. Rolux, "Efficient non-parametric function induction in semi-supervised learning," in *Proc. 10th Int. Workshop Artif. Intell. Statist.*, Hasting, Barbados, Jan. 2005, pp. 96–103.
[39] A. L. Neuenschwander, "Remote sensing of vegetation dynamics in response to flooding and fire in the Okavango Delta, Botswana," Ph.D. dissertation, Dept. Geography Environ., Univ. Texas, Austin, TX, USA, 2007.
[40] L. K. Saul and S. T. Roweis, "Think globally, fit locally: Unsupervised learning of nonlinear manifolds," *J. Mach. Learn. Res.*, vol. 4, pp. 119–155, Jun. 2003.
[41] A. B. Goldberg, X. Zhu, A. Singh, Z. Xu, and R. Nowak, "multi-manifold semisupervised learning," in *Proc. 12th Int. Conf. Artif. Intell. Statist.*, Clearwater Beach, FL, USA, 2009, vol. 5, pp. 169–176.
[42] H. B. Huang, H. Huo, and T. Fang, "Hierarchical manifold learning with applications to supervised classification for high-resolution remotely sensed images," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 3, pp. 1677–1692, Mar. 2014.
[43] Z. Y. Zhang, J. Wang, and H. Y. Zha, "Adaptive manifold learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 2, pp. 253–265, Feb. 2012.



Li Ma (M'13) received the B.S. and M.S. degrees from Shandong University, Jinan, China, in 2004 and 2006, respectively, and the Ph.D. degree in pattern recognition and intelligent system from Huazhong University of Science and Technology, Wuhan, China, in 2011.

During 2008–2010, she was a Visiting Scholar with Purdue University, West Lafayette, IN, USA. She is currently an Assistant Professor with the Faculty of Mechanical and Electronic Information, China University of Geosciences, Wuhan. Her research interests include hyperspectral data analysis, pattern recognition, and remote sensing applications.

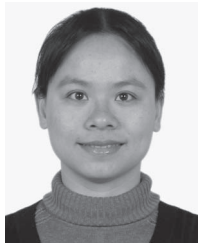
Melba M. Crawford (M'89–SM'05–F'07) received the B.S. and M.S. degrees in civil engineering from the University of Illinois at Urbana–Champaign, Champaign, IL, USA, in 1970 and 1973, respectively, and the Ph.D. degree in systems engineering from The Ohio State University, Columbus, OH, USA, in 1981.

She is a Professor of civil engineering, electrical and computer engineering, and agronomy with Purdue University, West Lafayette, IN, USA, where she is the Director of the Laboratory for Applications of Remote Sensing and the Chair of Excellence in Earth Observation. She also serves as the Associate Dean for Research of the College of Engineering. She has more than 150 publications in scientific journals, conference proceedings, and technical reports. Her research focuses on methods for analysis of multisensor remotely sensed data, including classification, unmixing, active learning, and knowledge transfer, and the application of these methodologies.

Dr. Crawford is the current President of the IEEE Geoscience and Remote Sensing Society (GRSS) and an Associate Editor of the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING. She has also served as the GRSS Executive Vice President, Vice President for Conferences and Symposia, Vice President for Professional Activities, and Education Director.

Xiaoquan Yang (M'13) received the B.S. degree from Shandong University, Jinan, China, in 2004 and the Ph.D. degree in biomedical engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2010.

From 2010 to 2012, he was a Postdoctoral Fellow with the Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology. He was promoted to Associate Professor in 2012. His research interests focus on optical imaging in biomedicine, pattern recognition, and medical image processing.



Yan Guo received the B.S., M.S., and Ph.D. degrees from the China University of Geosciences, Wuhan, China, in 1998, 2003, and 2009, respectively.

During 2012–2013, she was a Visiting Scholar with George Mason University, Fairfax, VA, USA. She is currently an Associate Professor with the College of Computer Science, China University of Geosciences. Her research interests include intelligence computation and remote sensing applications.