remote sensing

MDPI

Article

Deep Fully Convolutional Embedding Networks for Hyperspectral Images Dimensionality Reduction

Na Li [1], Deyun Zhou [1], Jiao Shi [1,*], Mingyang Zhang [2], Tao Wu [1] and Maoguo Gong [2]

[1] School of Electronics and Information, Northwestern Polytechnical University, 127 West Youyi Road, Xi'an 710072, Shaanxi, China; linaflydream@mail.nwpu.edu.cn (N.L.); dyzhounpu@nwpu.edu.cn (D.Z.); tao_woe@mail.nwpu.edu.cn (T.W.)

[2] Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education, Xidian University, Xi'an 710071, Shaanxi, China; myzhang@xidian.edu.cn (M.Z.); gong@ieee.org (M.G.)

* Correspondence: jiaoshi@nwpu.edu.cn

Abstract: Due to the superior spatial–spectral extraction capability of the convolutional neural network (CNN), CNN shows great potential in dimensionality reduction (DR) of hyperspectral images (HSIs). However, most CNN-based methods are supervised while the class labels of HSIs are limited and difficult to obtain. While a few unsupervised CNN-based methods have been proposed recently, they always focus on data reconstruction and are lacking in the exploration of discriminability which is usually the primary goal of DR. To address these issues, we propose a deep fully convolutional embedding network (DFCEN), which not only considers data reconstruction but also introduces the specific learning task of enhancing feature discriminability. DFCEN has an end-to-end symmetric network structure that is the key for unsupervised learning. Moreover, a novel objective function containing two terms—the reconstruction term and the embedding term of a specific task—is established to supervise the learning of DFCEN towards improving the completeness and discriminability of low-dimensional data. In particular, the specific task is designed to explore and preserve relationships among samples in HSIs. Besides, due to the limited training samples, inherent complexity and the presence of noise in HSIs, a preprocessing where a few noise spectral bands are removed is adopted to improve the effectiveness of unsupervised DFCEN. Experimental results on three well-known hyperspectral datasets and two classifiers illustrate that the low dimensional features of DFCEN are highly separable and DFCEN has promising classification performance compared with other DR methods.

Keywords: deep fully convolutional embedding network; dimensionality reduction; hyperspectral images; classification

Citation: Li, N.; Zhou, D.; Shi, J.; Zhang, M.; Wu, T.; Gong, M. Deep Fully Convolutional Embedding Networks for Hyperspectral Images Dimensionality Reduction. *Remote Sens.* **2021**, *13*, 706. https://doi.org/10.3390/rs13040706

Academic Editor: Jon Atli Benediktsson
Received: 31 December 2020
Accepted: 6 February 2021
Published: 15 February 2021

1. Introduction

With the rapid development of modern technology, hyperspectral imaging technology has been widely used in many fields, such as geology [1], ecology [2], geomorphology [3], atmospheric science [4], forensic science [5] and so on, not just in remote sensing satellite sensors and airborne platforms. Hyperspectral sensors can capture hundreds of narrow continuous spectral bands from visible to infrared wavelengths that are reflected or emitted from the scene. The 3D hyperspectral images (HSIs) have high spectral resolution and fine spatial resolution for the taken scene. These allow us to get more information about the object being studied. However, due to the high spectral dimensionality, the interpretation and analysis of hyperspectral images face many challenges. (1) Radiometric noise in some bands limits the precision of image processing [6]. (2) Some redundant bands reduce the quality of image analysis since the adjacent spectral bands are often correlated and not all bands are valuable for image processing [7]. (3) These redundant bands also lead to the cost of huge computational resources and storage space [8]. (4) There is a Hughes phenomenon, that is, the higher the data dimensionality, the poorer the classification performance because

of the limited samples [9]. These makes dimensionality reduction (DR) become an essential task for hyperspectral image processing.

Many classic algorithms have been used for HSIs DR, such as principal component analysis (PCA) [10], Laplacian eigenmaps (LE) [11], locally linear embedding (LLE) [11], Isometric feature mapping (ISOMAP) [12], linear discriminant analysis (LDA) [13]. These classical algorithms based on different concepts all attempt to explore and maintain the relationship among samples in HSIs, which is beneficial to improve the separability of low-dimensional features. However, there are several problems when they are applied for HSIs DR. Firstly, ISOMAP, LE and LLE have the out-of-sample problems. On this issue, locality preserving projection (LPP) [14] and neighborhood preserving embedding (NPE) [15] are proposed. Nevertheless, LPP, NPE, PCA and LAD are the linear transformations, which are ill-suited for HSIs because HSIs derived from the complex light scattering of natural objects are inherently nonlinear [16]. Also, spatial feature extraction is a common problem faced by these classical algorithms for HSI DR, which has allowed for good improvements in HSIs representation. Moreover, these algorithms focus on the shallow features of HSIs via a single mapping but cannot extract the deep complex features iteratively.

In recent years, deep learning, as one of the most popular learning algorithms, has been applied to various fields, which can yield more non-linear and more abstract deep representations of data by multiple processing layers [17]. The spatial features extraction is generally achieved by convolutional neural networks (CNN) which can exploit a set of trainable filters to capture local spatial features from receptive fields but often needs supervised information. Many studies have used CNN for HSIs [18]. Paoletti et al. [19] proposed a new deep convolutional neural network for fast hyperspectral image classification. Zhong et al. [20] proposed a supervised spectral-spatial residual network for HSIs on basic of the 3D convolutional layers. Han et al. [21] proposed a different-scale two-stream convolutional network for HSIs. These CNN-based methods can extract superior hyperspectral image features for classification, but they generally require enough class label samples for supervised learning. As a matter of fact, the task of labeling each pixel contained in HSIs is arduous and time-consuming, which generally requires a human expert. As a result, the class label samples of HSIs are scarce and limited, and even unavailable in some scenarios. To address this issue, a few of unsupervised CNN-based methods have been proposed for HSIs. Mou et al. [22] proposed a deep residual conv-deconv network for unsupervised spectral-spatial feature learning. Zhang et al. [23] proposed a novel modified generative adversarial network for unsupervised feature extraction in HSIs. Recently, Zhang et al. [24] proposed a symmetric all convolutional neural-network-based unsupervised feature extraction for HSIs. However, these unsupervised CNN-based approaches are usually based on data reconstruction, but they are short of the exploration of discriminability which is usually the primary goal of DR.

To overcome the drawbacks mentioned above, we propose an unsupervised deep fully convolutional embedding network (DFCEN) for dimensionality reduction of HSIs. Different from the conventional CNN-based network, DFCEN utilizes the learning parameters of convolutional (deconvolutional) layer to replace the fixed down-sampling (up-sampling) of pooling layer to improve the validity of the representation. Meanwhile parameter sharing of convolutional layer is conducive to the extraction of spatial features and reduce the number of parameters compared with fully-connected layer. For the convenience of explanation, DFCEN can be divided into two parts: convolutional subnetwork that encodes high-dimensional data into a low-dimensional space and deconvolutional subnetwork that recovers low-dimensional features to the original high-dimensional data. Accordingly, the network structure of DFCEN lays a foundation for unsupervised learning.

To address the shortcoming of the above unsupervised CNN-based approaches, we introduce a specific learning task of enhancing feature discriminability into DFCEN. Considering the completeness and discriminability of low-dimensional data, we particularly design a novel objective function containing two terms: reconstruction term and embedding term of the specific learning task. The former makes the low-dimensional features

keep completeness and original intrinsic information in HSIs. How to design a specific learning task to enhance the discriminability and separability of low-dimensional features is the key point of the latter. The relationships among samples is of considerable value, which are concerned in the classical DR algorithms described above and has been shown to be conducive to HSIs DR. In this paper, the DR concepts of two classical algorithms, LLE and LE, are used as references for the specific learning task in embedding term. Furthermore, in order to balance the contribution of two terms to DR, an adjustable trade-off parameter is added to the objective function. In addition, in order to reduce the training time, we choose to utilize the convolutional autoencoder (CAE) for pretraining to get good initial learning parameters of DFCEN.

Specifically, the contributions of this paper are as follows.

- An end-to-end symmetric fully convolutional network, DFCEN, is proposed for HSIs DR, which is the foundation of unsupervised learning. In addition, owing to the symmetry of DFCEN, the network structure of symmetry layer in convolutional subnetwork and deconvolutional subnetwork is the same. For that, these two subnetwork can share the same pretraining parameters, which saves the pretraining time.
- A novel objective function with two terms constraining different layers respectively is designed for DFCEN. This allows DFCEN to explore not only completeness but also discriminability compared to the previous unsupervised CNN-based approaches
- This is the first work to introduce LLE and LE into an unsupervised fully convolutional network, which simultaneously solved their out-of-sample, linear transformation, and spatial feature extraction problem. In addition, other different DR concepts also can be implemented in embedding term as long as it can be expressed in the form of an objective function.
- Due to the limited training samples, inherent complexity and the presence of noise bands in HSIs, DFCEN as an unsupervised network is sensitive to input data. So, a preprocessing strategy of removing noise band is adopted, which is proved to effectively improve the DFCEN representation of HSIs.

This paper is organized as follows. In Section 2, we introduce the background and the related works. The proposed deep fully convolutional embedding network are described in detail in Section 3. Section 4 presents the experimental results on three datasets that demonstrate the superiority of the proposed DR method. A conclusion is presented in Section 5.

2. Background and the Related Works

2.1. Mutual Information

Mutual information (MI) has the capacity of measuring the statistical dependence between two random variables [25]. Treating spectral bands and Ground Truth map G shown in Figures 7b, 8b and 9b as random variables, MI can be used to evaluate the relative utility of each band to classification [8]. Given two random variables a and b with marginal probability distributions $p(a)$ and $p(b)$ and joint probability distribution $p(a,b)$, the MI is defined as below

$$\mathrm{MI}(a,b) = \sum_{a\in \mathrm{a}, b\in \mathrm{b}} p(a,b) \log \frac{p(a,b)}{p(a)\cdot p(b)}. \tag{1}$$

The higher the MI value between a band and G, the greater the contribution of this band to classification. In practical application, G usually cannot be obtained. The work [8] used an estimated ground truth map $\widehat{G} = \frac{1}{|E|}\sum_{I_j\in E} I_j$ to evaluate the contribution of each band to classification. I_j is a spectral band and E is a set of bands with the highest entropy. Let random variable a take values in the set a with the probability distribution $p(a)$, the entropy is defined by $H(a) = -\sum_{a\in \mathrm{a}} p(a)\log p(a)$ [26].

2.2. Locally Linear Embedding

Locally linear embedding (LLE) is an unsupervised learning algorithm that computes low-dimensional, neighborhood-preserving embeddings of high-dimensional inputs [27]. The local geometry is characterized by linear coefficients that reconstruct the data point using its neighbors [28]. For a data set $X = \{x_1, x_2, ..., x_i, ..., x_m\}$, assuming that x_i can be reconstructed by a linear combination of neighborhood samples x_k, x_l, x_s, that is $x_i = w_{ik}x_k + w_{il}x_l + w_{is}x_s$, the low-dimensional data also maintains the same linear relationship which is $z_i = w_{ik}z_k + w_{il}z_l + w_{is}z_s$. The linear reconstruction coefficients are obtained by the following optimization

$$\begin{array}{l} \min\limits_{w_{ij}} \sum_{i=1}^{m} \left\| x_i - \sum_{j \in Q_i} w_{ij} x_j \right\|_2^2 \\ \text{s.t.} \sum_{j \in Q_i} w_{ij} = 1 \end{array}, \tag{2}$$

where Q_i is a sample set consisting of the nearest k neighbor samples of x_i based on the Euclidean distance. The coefficient w_{ij} has a closed solution

$$w_{ij} = \frac{\sum_{h \in Q_i} C_{jh}^{-1}}{\sum_{l,s \in Q_i} C_{ls}^{-1}}, \tag{3}$$

where $C_{ij} = \left(x_i - x_j\right)^T (x_i - x_k)$. w_{ij} summarizes the contribution of x_j to the reconstruction of x_i. According to LLE, the extracted features should preserve neighborhood geometric manifold [29], therefore the embedding cost function is

$$\begin{cases} \min\limits_{z_1,...,z_m} \sum_{i=1}^{m} z_i - \sum_{j \in Q_i} w_{ij} z_j \Big\|_2^2 \\ \text{s.t.}\ Z = A^T X, \sum_{i=1}^{n} z_i = 0, \frac{1}{n} A A^T = I \end{cases} \tag{4}$$

where z_i is the low-dimensional data point corresponds to x_i. $Z = \{z_1, z_2, ..., z_m\}$ is the low-dimensional representation. LLE maps its inputs into a single global coordinate system of lower dimensionality. LLE explores the reconstructed relationship between each sample and its nearest neighbors, preserving the manifold structure of the data.

2.3. Laplacian Eigenmaps

Laplacian Eigenmaps [11] (LE) has remarkable properties of preserving local neighborhood structure of data. LE is to construct the relationship between data with local angles and reconstruct the local structure and features of the data by constructing adjacency graph [30]. If two data instances x_i and x_j are very similar, i and j should be as close as possible in the target subspace after dimensionality reduction. Its intuitive concept is to hope that the points that are related to each other (the points connected in the graph) are as close as possible in the low-dimensional space.

A k-nearest neighborhood graph or an ε-ball neighborhood graph is constructed and weights of edges (between vertices) are assigned using the Gaussian kernel function or 0–1 weighting method [31]. Given a dataset $X = \{x_1, x_2, ..., x_n\}$ with n samples, each sample $x_i \in X$ has m features. Let $y_1, y_2, ..., y_n$ be the d dimensional representations of X. That is, each y_i is a d dimensional row vector. With LE, the lower dimensional representation of X can be achieved by solving the following optimization problem

$$\min_{y_1, y_2, ..., y_n} \sum_{i}^{n} \sum_{j}^{n} \left\| y_i - y_j \right\|^2 M_{ij}, \tag{5}$$

where $M = (M_{ij})_{n\times n}$ is the weight matrix of the k-nearest neighborhood graph. The weight matrix M is calculated based on the Euclidean distance between samples, which is defined as

$$M_{ij} = \begin{cases} e^{-\frac{\|x_i - x_j\|^2}{t}}, & x_j \in Q_i \\ 0, & x_j \notin Q_i \end{cases}, \tag{6}$$

where Q_i is a sample set consisting of the nearest k neighbor samples of x_i based on the Euclidean distance. t is an adjustable parameter. LE explores and preserves the relationship between each sample and its nearest neighbors.

2.4. Convolutional Autoencoder

Convolutional autoencoder adopts the convolutional layer instead of the full-connected layer, of which the principle is the same as the autoencoder [32]. Figure 1 shows the structure of the 2D convolutional autoencoder which comprises an encoder and a decoder. The encoder encodes the input data and maps the features to the hidden layer space, and then the decoder decodes the features of the hidden layer space (the process of reconstruction) to obtain the reconstructed samples of the input [33]. For a input data $X \in \Re^{s1\times s1\times d1}$, the encoder is defined as

$$h = \mathrm{s}(\mathrm{conv2}(X, \theta)), h \in \Re^{s2\times s2\times d2}, \tag{7}$$

where conv2() represents the 2D convolution and θ is the learning parameter in the encoder. h is the output of the hidden layer in the 2D convolutional autoencoder and s() is the activation function. Based on h, the decoder is defined as

$$X' = \mathrm{s}\left(\mathrm{dconv2}\left(h, \theta'\right)\right), X' \in \Re^{s1\times s1\times d1}, \tag{8}$$

where dconv2() represents the 2D deconvolution and θ' is the learning parameter in the decoder. X' stands for the output of the reconstruction layer and has the same structure as the input data X. The cost function can be defined as

$$L(X; \theta, \theta') = \left\|X - X'\right\|^2. \tag{9}$$

Compared with the traditional autoencoder, the convolutional encoder is more advantageous in extracting spatial features from images [34].

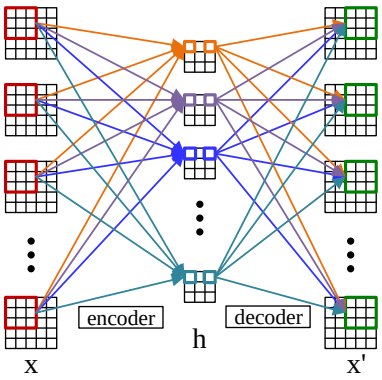


Figure 1. The structure of the 2D convolutional autoencoder.

3. The Proposed Method

In this section, we will introduce our proposed method in detail. The flowchart is shown in Figure 2. Usually due to changes in atmospheric conditions, occlusion caused by the presence of clouds, changes in lighting, and other environmental disturbances, some noise bands in HSIs increase the difficulty in feature extraction and classification. As an

unsupervised network, DFCEN is sensitive to these noise spectral bands because of the limited training samples and complex intrinsic features of HSIs. For this reason, a simple band selection based on mutual information is adopted for selecting and removing the noise bands at first. Then the relationships among samples is obtained for the specific learning task, which is specially based on LLE and LE in this paper. Next, training samples specifically applied to DFCEN are generated through a data preprocessing. Afterwards, DFCEN is learning from the training samples and relationship among samples. Eventually, the low-dimensional features from DFCEN is classified by classifiers.

3.1. Data Preprocessing

Data preprocessing includes data standardization, data denoising and data expansion. Data standardization is to standardize the pixel values of each spectral band to 0∼1 since it is not appropriate to directly process the raw HSIs data with large pixel values. Data denoising is to select and remove the noise spectral band that may disturb feature extraction and classification. MI can evaluate the contribution of each band to classification [8], Besides, due to the simplicity of calculation, MI is adopted to search for bands that contribute little to the classification as the noise spectral band. Each band I_j in HSIs is considered as a random variable. Its probability distribution function can be estimated as $p(I_j) = \frac{h(I_j)}{m \times n}$, where $h(I_j)$ represents the gray-level histogram of the jth band with $m \times n$ pixels. The joint probability distributions of any two bands in HSIs is estimated by $p(I_i, I_j) = \frac{H(I_i, I_j)}{m \times n}$, where $H(I_i, I_j)$ is the joint gray-level histogram of the ith and jth band.

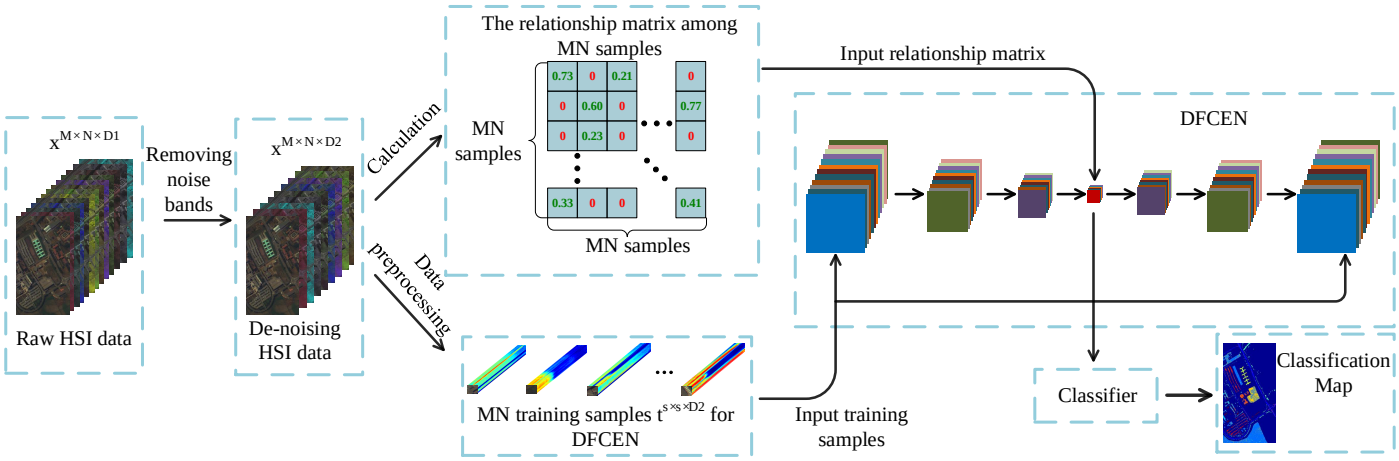
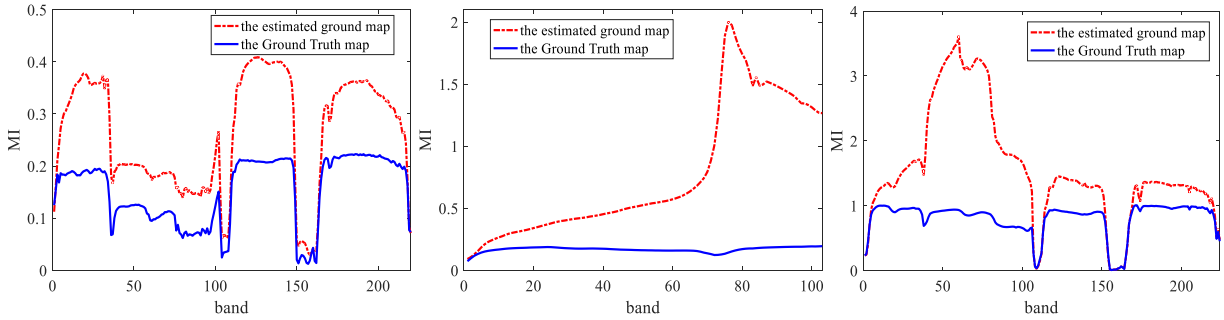


Figure 2. Flowchart of the proposed method.

Figure 3 shows the MI values of each band in three datasets. As we can see, the two lines fluctuate almost identically. For this reason, we can find and remove noise bands with low MI in an unsupervised way according to the red dotted line. For a raw HSIs data $X \in \Re^{M\times N\times D1}$, where M and N is the spatial size and $D1$ is the raw number of the spectral bands, the corresponding de-noising data can be expressed as $X \in \Re^{M\times N\times D2}$, where $D2$ is the number of bands after removing the noise bands and $D2 < D1$. Actually, we only removed 30 noise bands for Indian Pines dataset, 0 band for Pavia University dataset, 8 bands for Salinas dataset. In order to further prove the validity of removing the noise bands before DFCEN, we take the Indian Pines dataset as an example to compare the classification accuracy of different dimensionality reduction algorithms before and after removing the noise bands. From Table 1, NBS means that the algorithm directly acts on the raw data while BS represents removing the noise bands before dimensionality reduction algorithm. It can be seen from Table 1 that for two unsupervised methods based on neural network, DFCEN and SAE, removing the noise bands is conducive to improving classification accuracy. In the meantime, it also slightly improves other dimensionality reduction algorithms.



(**a**) Indian Pines (**b**) Pavia University (**c**) Salinas

Figure 3. MI values of each spectral band with the Ground Truth map and the estimated ground map on three datasets.

Table 1. Classification accuracy of different dimensionality reduction (DR) algorithms with or without band selection for Indian Pines dataset.

		RAW	LE	LLE	SAE	LPNPE	SSRLDE	SSMRPE	SSLDP	DFCEN_LE	DFCEN_LLE
SVM	NBS	76.1	78.0	61.3	75.0	86.2	83.1	78.9	74.1	88.4	89.9
	BS	83.1	78.1	60.5	81.6	85.3	84.6	81.1	75.7	90.3	91.7
KNN	NBS	68.7	77.5	57.6	61.4	85.4	81.6	77.7	72.5	85.0	87.5
	BS	72.4	77.2	68.8	70.5	80.5	78.9	74.7	67.1	86.9	89.3

Spatial features have been proven to be beneficial to improve the representation of HSIs and increase interpretation accuracy [35,36]. For each pixel, the neighborhood pixel is one of the most important spatial information which is fed to DFCEN in the form of neighborhood window centered around each pixel. With this in mind, the input data size of DFCEN is designed as $s \times s \times D2$, where s is the size of the neighborhood window and $D2$ is the number of bands. However, the problem is that the neighborhood window of the pixels at the image boundary is incomplete. These boundary pixels cannot be ignored since our goal is to reduce the dimensions of each pixel in HSIs. It is also inappropriate to simply fill the neighborhood window of boundary pixels with 0. In order to deal with this problem better, we implement a data expansion strategy based on the Manhattan distance to fill the neighborhood window of the boundary pixels. Figure 4 shows the process of expanding the data by two layers, where the dark color is the original data and the light color is the filling data. For a pixel $p^{1\times D2}$ in a de-noising HSI $x^{MN\times D2}$ (MN is the number of pixels), its neighborhood window is a training sample $t^{s\times s\times D2}$ that is fed to the proposed DFCEN. As a result, a training sample set $T^{s\times s\times D2\times MN}$ with MN samples can be generated from a de-noising HSI $x^{M\times N\times D2}$.

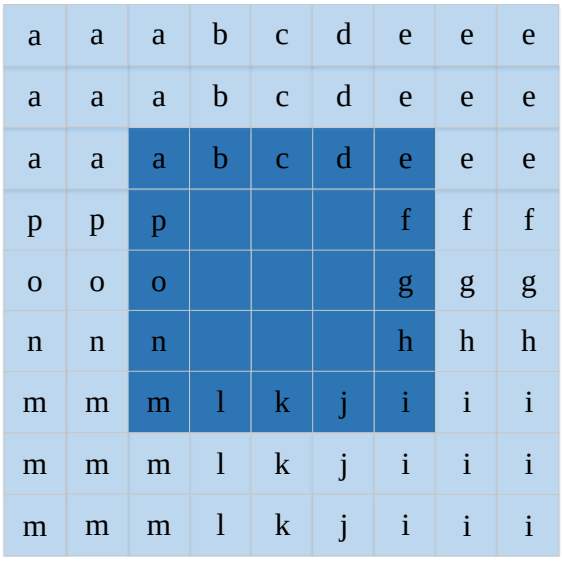


Figure 4. Data expansion strategy. This is the data expansion process when the size of neighborhood window is 5.

3.2. Structure of DFCEN

DFCEN is composed of convolutional layer and deconvolutional layer, excluding pooling layer and full-connected layer. Accordingly, DFCEN can be divided into two parts: convolutional subnetwork and deconvolutional subnetwork. In the convolutional subnetwork, the input data is propagated through multiple convolutional layers to a perception layer, while this perception layer is propagated through multiple deconvolutional layers to a output layer (whose size is same as the input layer) in the deconvolutional subnetwork.

Figure 5 shows the network structure of DFCEN. The introduction in the red box is the name and structure of each layer, while the name of the learning parameter and the filter size is in the green box. It is worth emphasizing that DFCEN is a symmetric and end-to-end network where the number of layers can be set or changed based on specific data or tasks. For the sake of explanation, we take a 7-layer DFCEN shown in Figure 5 as an example to introduce the network structure characteristics of DFCEN in detail. The following is the description of a 7-layer DFCEN shown in Figure 5.

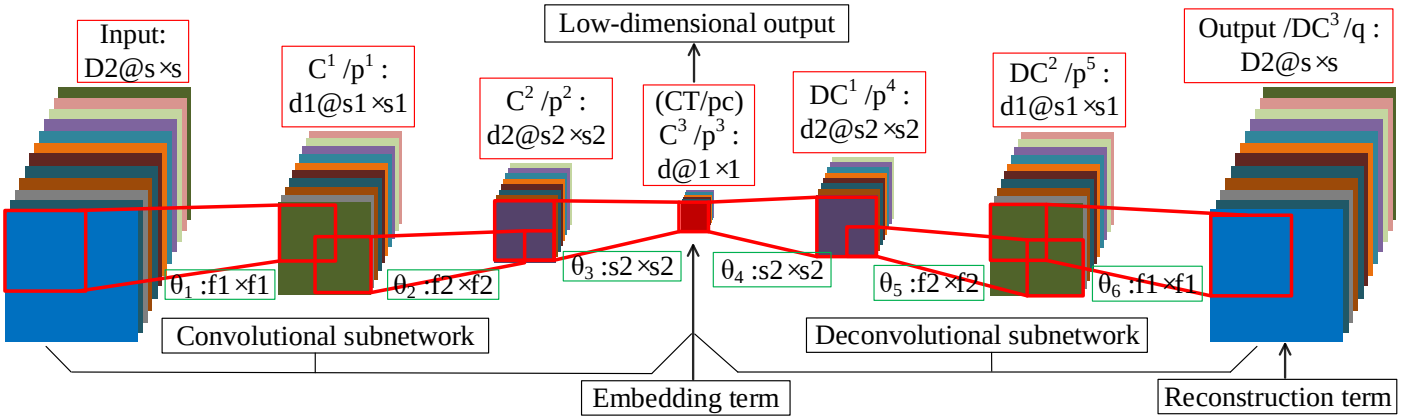


Figure 5. The structure of the proposed deep fully convolutional embedding network.

In the convolutional subnetwork, firstly, a training sample $t^{s\times s\times D2}$ is fed to DFCEN, where $D2$ is also the number of channels of the input layer. Secondly, the output of input layer is sent to the first convolutional layer C^1 through $d1$ filters of size $f1 \times f1$. The output of C^1 contains $d1$ feature maps $\left(p^1\right)^{s1\times s1\times d1}$ that are then transmitted to the second convolutional layer C^2 via $d2$ filters of size $f2 \times f2$. Next, $d2$ feature maps $\left(p^2\right)^{s2\times s2\times d2}$ are

obtained after C^2 is activated, which are then send to the last convolutional layer C^3 by d filters of size $s2 \times s2$. The last convolutional layer in the convolutional subnetwork is also the central layer CT of the whole DFCEN. Eventually, a low-dimensional feature $pc^{1\times1\times d}$ of concern is generated after applying the activation function to CT.

In the deconvolutional subnetwork, the low-dimensional feature $pc^{1\times1\times d}$ (which is also the output of the convolutional subnetwork) from CT is up-sampled layer by layer through multiple deconvolutional layers. At first, $pc^{1\times1\times d}$ is sent to the first deconvolutional layer DC^1 with $d2$ filters of size $s2 \times s2$. Then, $d2$ feature maps $\left(p^4\right)^{s2\times s2\times d2}$ are gained after the activation function and then transfered to the second deconvolutional layer DC^2 through $d1$ filters of size $f2 \times f2$. Next, after activating DC^2, $d1$ feature maps $\left(p^5\right)^{s1\times s1\times d1}$ are obtain and transfered to the last deconvolutional layer DC^3 (which is also the output layer of the whole DFCEN) with $D2$ filters size of $f1 \times f1$. In the end, the output $q^{s\times s\times D}$ of the whole DFCEN is generated after DC^3 is activated, whose size is the same as the input of DFCEN.

In fact, the characteristics of DFCEN are the size and number of filters (learning parameters), which are identical for the symmetrical layer in the convolutional and deconvolutional subnetwork. This rule also applies to the number and size of feature maps per layer. In particular, the number of feature maps per layer exists: $D2 \geq d1 \geq d2 \geq d$ where d is target dimension of dimensionality reduction and $D2$ is the dimension of input data. Meanwhile, the relationship of the size of feature maps per layer is $s \geq s1 \geq s2 \geq 1$ where s is the size of input data and the size of CT must be 1 since it represents the low-dimensional features of one pixel. For this reason, the size of the filter between CT and its preceding layer must be the same as the size of its preceding layer. In Figure 5, the preceding layer of CT is C^2. In brief, DFCEN is a symmetric full convolutional network with a central layer of size 1, where the convolutional subnetwork reduce the dimensionality and size of data layer by layer while the deconvolutional subnetwork restores the data dimensionality and size layer by layer. Therefore, the network structure determines that feature extraction of DFCEN is an unsupervised process as long as the embedding term in objective function does not require any class label information.

3.3. Objective Function of DFCEN

As discussed in Section 1, DFCEN supports not only unsupervised feature extraction based on data reconstruction, but also task-specific learning which is conducive to dimensionality reduction and classification. The objective function of DFCEN consists of two terms: embedding term for the specific learning task and reconstruction term. The embedding term can be changed or designed according to specific concept or task, which is dedicated to improving the discriminant ability of the low-dimensional features. As shown in Figure 5, the embedding term is to constrain the low-dimensional output of the central layer CT. So it only acts on the parameter update of the convolutional subnetwork. For a training sample set $T^{s\times s\times D2\times MN} = \{t_1, t_2, ..., t_i, ..., t_{MN}\}, t_i \in \Re^{s\times s\times D2}$, the output of CT in Figure 5 is expressed as follows

$$pc(t_i, \Theta_d) = \mathrm{s}(\mathrm{conv2}(\mathrm{s}(\mathrm{conv2}(\mathrm{s}(\mathrm{conv2}(t_i, \theta_1)), \theta_2)), \theta_3)), \tag{10}$$

where $\Theta_d = \{\theta_1, \theta_2, \theta_3\}$ is the learning parameters in the convolutional subnetwork. $\mathrm{conv2}()$ denotes the 2D convolution and $\mathrm{s}()$ is the activation function $\mathrm{s}(x) = \log(1 + e^x)$. $pc(t_i, \Theta_d)$ is also the low-dimensional representation of DFCEN.

In order to enhance the separability and discriminability of low-dimensional features, we explore and maintain the relationship among samples as a specific learning task. In this paper, LLE and LE, two classical manifold learning algorithms are introduced into the embedding term of DFCEN.

3.3.1. LLE-Based Embedding Term

LLE aims at preserving the original reconstruction relationship between each sample and its neighbors in the mapping space, which assumes that a sample data can be reconstructed by a linear combination of its neighborhood samples. The linear reconstruction is described in Equation (2). The original reconstruction coefficient W can be calculated according to Equation (3). For a HSI dataset $x^{M\times N\times D2}$, the relationship coefficient W can be expressed as: $W^{MN\times MN} = \{w_{11}, w_{12}, ..., w_{ij}, ..., w_{MN\times MN}\}$. Since the coefficient W only characterizes the relationship between the sample and its nearest k neighbor samples, it can also be described as

$$w_{ij} = \begin{cases} \dfrac{\sum_{h\in Q_i}\left[(x_i-x_j)^T(x_i-x_h)\right]^{-1}}{\sum_{l,s\in Q_i}\left[(x_i-x_l)^T(x_i-x_s)\right]^{-1}}, & x_j \in Q_i \\ 0, & x_j \notin Q_i \end{cases}. \tag{11}$$

Q_i is the nearest k neighbor samples of x_i. The number of selected neighbor samples k is much smaller than the total number of samples MN, namely, $k \ll MN$. Therefore, the relationship coefficient matrix W is a sparse matrix.

Referring to LLE, the embedding term should constrain the low-dimensional representation to maintain the original reconstruction relationship. Hence, for a training sample set $T^{s\times s\times D2\times MN}$, the LLE-based embedding term can be defined as follow

$$L_{\mathrm{ED\_LLE}}(T,\Theta_d) = \min_{\Theta_d} \tfrac{1}{MN}\textstyle\sum_{i=1}^{MN} \|pc(t_i,\Theta_d) - \sum_{j=1}^{MN} w_{ij} pc(t_j,\Theta_d)\|_F^2, \tag{12}$$

where w_{ij} is the original reconstruction coefficient that is calculated according to Equation (11), which is a constant for the LLE-based embedding term. $pc(t_i,\Theta_d)$ is the output of CT in DFCEN. Θ_d is the learning parameters in the convolutional subnetwork. MN is the number of training samples in T. $\|\cdot\|_F^2$ is the square of the F norm, which is to calculate the sum of the squares of all the elements inside.

3.3.2. LE-Based Embedding Term

LE is to construct the relationship among samples with local angels and reconstruct the local structure and features in the low-dimensional space. An adjacency graph based on the Euclidean distance is constructed to characterize the relationship among samples, which is also called the weight matrix and defined in Equation (6). When the sample x_j does not belong to the nearest k neighbor samples of the sample x_i , the weight coefficient M_{ij} between the samples x_j and x_i is 0. In fact, for a HSI dataset $x^{M\times N\times D2}$, due to $k \ll MN$, the adjacency graph matrix M is also a sparse matrix. In practice, LE hopes that samples that are related to each other (the points connected in the adjacency graph) are as close as possible in the low-dimensional space, which is described in a formula in Equation (5).

Referring to LE, for samples that are related in the original space, the embedding term should constrain their low-dimensional representation as close as possible. As a result, for a training sample set $T^{s\times s\times D2\times MN}$, the LE-based embedding term can be defined as follow

$$L_{\mathrm{ED\_LE}}(T,\Theta_d) = \min_{\Theta_d} \tfrac{1}{MN}\textstyle\sum_{i=1}^{MN}\sum_{j=1}^{MN} \|pc(t_i,\Theta_d) - pc(t_j,\Theta_d)\|_F^2 M_{ij}, \tag{13}$$

where M_{ij} is the adjacency graph coefficient in the original space, which also is a constant.

3.3.3. Reconstruction Term

As shown in Figure 5, the reconstruction term is to constrain the output of the whole DFCEN. So it acts on all learning parameter updates. The reconstruction term ensures

that low-dimensional features can be restored as input data. For a training sample set $T^{s\times s\times D\times MN}$, the output of DFCEN in Figure 5 is expressed as follow

$$q(t_i,\Theta) = \mathrm{s}(\mathrm{dconv2}(\mathrm{s}(\mathrm{dconv2}(\mathrm{s}(\mathrm{dconv2}(pc(t_i,\Theta_d),\theta_4)),\theta_5)),\theta_6)), \tag{14}$$

where $\Theta = \{\Theta_d,\theta_4,\theta_5,\theta_6\}$ represents all learning parameters in DFCEN and $\{\theta_4,\theta_5,\theta_6\}$ is the parameters in the deconvolutional subnetwork. dconv2() denotes the 2D deconvolution and s() is the activation function. $pc(t_i,\Theta_d)$ is the output of the convolutional subnetwork.

The reconstruction term aims at maintaining original intrinsic information, which restores the low-dimensional features to the original input data. After the low-dimensional representation pc is propagated by the multiple deconvolutional layers, the reconstructed data q is obtained. The reconstruction term minimizes the error between the reconstructed data and the original input data. For a training sample set $T^{s\times s\times D2\times MN}$, the reconstruction term can be described as follow

$$L_{\mathrm{RT}}(T,\Theta) = \min_{\Theta}\frac{1}{MN}\sum_{i=1}^{MN}\|t_i - q(t_i,\Theta)\|_F^2, \tag{15}$$

where $q(t_i,\Theta)$ is the output of DFCEN and Θ denotes all learning parameter.

3.3.4. Objective Function

The embedding and reconstruction term have been introduced above. The embedding term constrains the low-dimensional output of the central layer to maintain the original sample relationship, while the reconstruction term ensures that the low-dimensional feature is reconstructed back to the high-dimensional input data. To balance the effects of these two terms on dimensionality reduction, a trade-off parameter is added to the objective function. As a result, for a training sample set $T^{s\times s\times D2\times MN}$, the objective function of DFCEN can be described as

$$L(T,\Theta) = L_{\mathrm{RT}}(T,\Theta) + \lambda L_{\mathrm{ED}}(T,\Theta_d), \tag{16}$$

where λ is a adjustable trade-off parameter. $L_{\mathrm{RT}}(T,\Theta)$ is the reconstruction term and $L_{\mathrm{ED}}(T,\Theta_d)$ is the embedding term.

3.4. Learning of DFCEN

The learning of DFCEN is to optimize the network parameters Θ according to the objective function which is formulated in Equation (16). In this paper, we adopt the gradient descent method to optimize learning parameters. The update formula for Θ is expressed as $\Theta = \Theta - \Delta\Theta$, where $\Delta\Theta$ is the partial derivative of the objective function with respect to Θ, which has the form

$$\Delta\Theta = \frac{\partial L_{\mathrm{RT}}(T,\Theta)}{\partial\Theta} + \lambda\frac{\partial L_{\mathrm{ED}}(T,\Theta_d)}{\partial\Theta}. \tag{17}$$

In the following, we calculate these two partial derivatives separately. For a training sample t_i, the partial derivative from the reconstruction term can be formulated as

$$\frac{\partial}{\partial\Theta}L_{\mathrm{RT}}(t_i,\Theta) = \frac{\partial}{\partial\Theta}\|t_i - q(t_i,\Theta)\|_F^2 = \frac{\partial}{\partial\Theta}\mathrm{tr}((t_i - q(t_i,\Theta))^T(t_i - q(t_i,\Theta))) = 2(q(t_i,\Theta) - t_i)\frac{\partial q(t_i,\Theta)}{\partial\Theta}, \tag{18}$$

Here $\frac{\partial q(t_i,\Theta)}{\partial\Theta}$ is the partial derivative of the output layer (also last layer) with respect to all network parameters $\Theta = \{\theta_1,\theta_2,\theta_3,\theta_4,\theta_5,\theta_6\}$. For the 7-layer DFCEN shown in Figure 5, $\{\theta_1,\theta_2,\theta_3\}$ is the parameters in the convolutional subnetwork while $\{\theta_4,\theta_5,\theta_6\}$ is

in the deconvolutional subnetwork. For $\{\theta_1, \theta_2, \theta_3\}$, the partial derivative with respect to the lth layer parameters θ_l can be calculated as

$$\frac{\partial q(t_i, \theta_l)}{\partial \theta_l} = \text{rot180}(\text{conv2}(p^{l-1}, \text{rot180}(s'(L^l)))), \tag{19}$$

where p^{l-1} is the feature maps in the $(l-1)$th layer and L^l is the lth layer of DFCEN. When $l = 1$, p^{l-1} is the input data t_i. The derivation process can be consulted in [37]. rot180() represents a rotation of 180 degrees. conv2() is a 2D convolution. s′ is the derivative function of the activation function, which is described as $s'(x) = \frac{e^x}{1+e^x}$. For $\{\theta_4, \theta_5, \theta_6\}$, the partial derivative is calculated as

$$\frac{\partial q(t_i, \theta_l)}{\partial \theta_l} = \text{rot180}(\text{dconv2}(p^{l-1}, \text{rot180}(s'(L^l)))), \tag{20}$$

where dconv2() is a 2D deconvolution.

The embedding term is only responsible for updating the parameters $\Theta_d = \{\theta_1, \theta_2, \theta_3\}$ in the convolutional subnetwork. For a training sample t_i, the partial derivative of the LLE-based embedding term with respect to Θ_d can be formulated as

$$\begin{aligned} \frac{\partial}{\partial \Theta} L_{\text{ED\_LLE}}(t_i, \Theta_d) &= \frac{\partial}{\partial \Theta} \| pc(t_i, \Theta_d) - \sum_{j=1}^{MN} w_{ij} pc(t_j, \Theta_d) \|_F^2 \\ &= 2(pc(t_i, \Theta_d) - \sum_{j=1}^{MN} w_{ij} pc(t_j, \Theta_d)) \cdot (\frac{\partial pc(t_i, \Theta_d)}{\partial \Theta} - \sum_{j=1}^{MN} w_{ij} \frac{\partial pc(t_j, \Theta_d)}{\partial \Theta}), \end{aligned} \tag{21}$$

Here w_{ij} is a constant. $\frac{\partial pc(t_i, \Theta_d)}{\partial \Theta}$ is the partial derivative of the central layer CT with to the parameters Θ_d in the convolutional subnetwork. It can be expressed in the form of Equation (19). The partial derivative of the LE-based embedding term can be formulated as

$$\begin{aligned} \frac{\partial}{\partial \Theta} L_{\text{ED\_LE}}(t_i, \Theta_d) &= \frac{\partial}{\partial \Theta} \| pc(t_i, \Theta_d) - pc(t_j, \Theta_d) \|_2^2 M_{ij} \\ &= 2(pc(t_i, \Theta_d) - pc(t_j, \Theta_d))(\frac{\partial pc(t_i, \Theta_d)}{\partial \Theta} - \frac{\partial pc(t_j, \Theta_d)}{\partial \Theta}) M_{ij}, \end{aligned} \tag{22}$$

where M_{ij} is also a constant.

In order to reduce the training time, we choose to use the convolutional autoencoder (CAE) to pretrain network to obtain good initial parameters. Owing to the symmetry of DFCEN, the parameter structure between the layers in the convolutional subnetwork is the same as that between the corresponding layers in the deconvolutional subnetwork. For this reason, symmetrical layers of two subnetworks can be initialized with the same parameters. So, a 7-layer DFCEN shown in Figure 5 only requires 3 CAEs for pretraining parameters, which saves the pretraining time. Figure 6 shows the pretraining process, where only after the first CAE has been trained can the second CAE be trained, and so on. The parameters in Figure 6, corresponding to the parameters in Figure 5, initializes DFCEN. The activation function of CAE is the same as that of DFCEN.

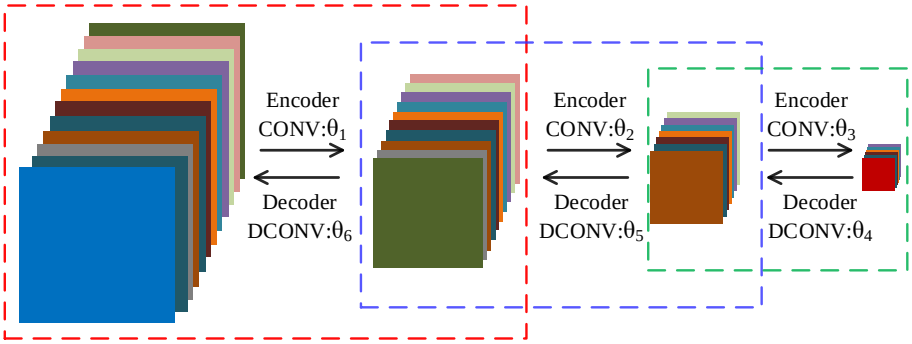


Figure 6. Pretraining process. Each dashed box represents a convolutional autoencoder. $\{\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6\}$ correspond to the parameters in Figure 5.

4. Experimental Study

4.1. Description of Data Sets

The first dataset, Indian Pines Dataset, covering the Indian Pines region, northwest Indiana, USA, was acquired by the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) sensor in 1992. The spatial resolution of this image is 20 m. It has 220 original spectral bands in the 0.4–2.5 µm spectral region and each band contains 145 × 145 pixels. Owing to the noise and water absorption, 20 spectral bands are abandoned and the remaining 200 bands are used in this data set. This dataset contains background with 10,776 pixels and 16 ground-truth classes with 10,249 pixels. The number of pixels in each class is range from 20 to 2455. The color image and the labeled image with 16 classes are shown in Figure 7.

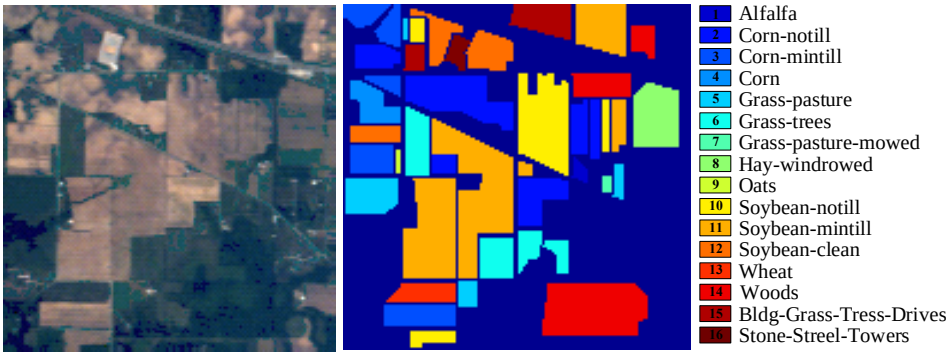


Figure 7. Indian Pines dataset: (**a**) the color image, (**b**) the Ground Truth map.

The second dataset covers the University of Pavia, Northern Italy, which was acquired by the Reflective Optics System Imaging Spectrometer (ROSIS) sensor and called Pavia University Dataset. Its spectral range is 0.4–0.82 µm. After removing 12 noise bands from the original dataset with 115 spectral bands, 103 bands are employed in this paper. The spatial resolution is 1.3 m and each band has 610 × 340 pixels. This dataset consists of 9 ground-truth classes with 42,776 pixels and background with 164,624 pixels. Figure 8 shows the color image and the labeled image with 9 classes.

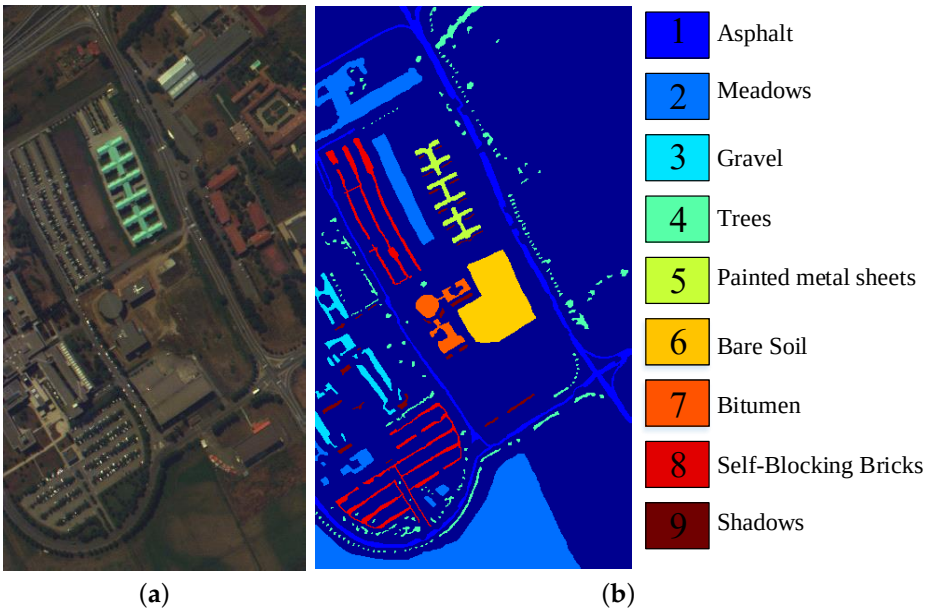


Figure 8. Pavia University dataset: (**a**) the color image, (**b**) the Ground Truth map.

The third dataset, Salinas Dataset, covering Salinas Vally, CA, was acquired by AVIRIS sensor in 1998, whose spatial resolution is 3.7 m. There are 224 original bands with spectral ranging from 0.4 to 2.45 μm. Each band has 512 × 217 pixels including 16 ground-truth classes with 56,975 pixels and background with 54,129 pixels. After removing 20 bands that are severely affected by noise, the remaining 204 bands are used for the experiments. The color image and the labeled image with 16 classes are shown in Figure 9.

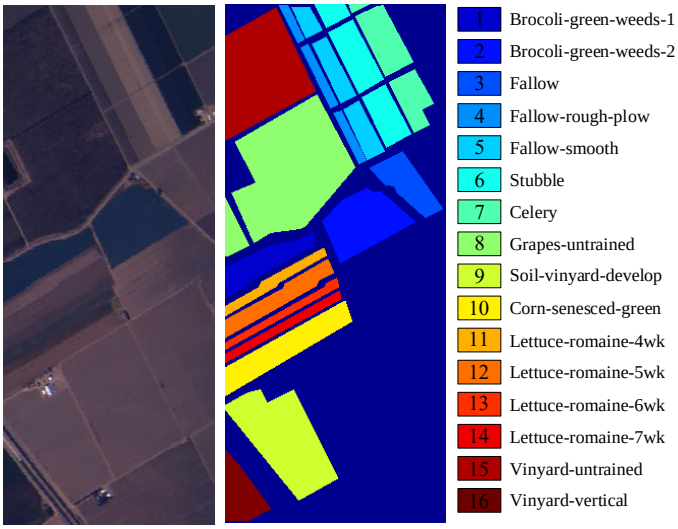


Figure 9. Salinas dataset: (**a**) the color image, (**b**) the Ground Truth map.

4.2. Experimental Setup

For the sake of clarity, the proposed DFCEN with LLE-based embedding term is named DFCEN_LLE below while that with LE-based embedding term is written as DFCEN_LE. The network structure of DFCEN for three datasets is experientially designed on the basis of the structure of DFCEN described in Section 3.2. In this paper, DFCEN_LLE and DFCEN_LE have the same network structure for experimental convenience. The following is the network structure with a target dimensionality of 30. For the Indian Pines dataset, the network structure is 170–100–50–30–50–30–170 and the size of filter per layer is 3 × 3–2 × 2–2 × 2–2 × 2–2 × 2–3 × 3. For the Pavia University dataset, the network structure is 103–70–30–70–30–103 and the size of filer in all layers is 3 × 3. For the Salinas dataset, the network structure is 196–110–60–30–60–110–196 and the size of filer per layer is also 3 × 3.

To prove the effectiveness, DFCEN is compared with several dimensionality reduction algorithms, such as LE [11], LLE [11], SAE, spatial-domain local pixel NPE (LPNPE) [38], spatial and spectral regularized local discriminant embedding (SSRLDE) [38], SSMRPE [39], spatial–spectral local discriminant projection (SSLDP) [40]. The former three methods are spectral-based methods while the latter four approaches make use of both spatial and spectral information for dimensionality reduction of HSIs. Besides, the raw HSIs is also used for comparison. SAE is a algorithm based on neural network, and its network structures are 170–100–50–30–170 for Indian Pines dataset, 103–70–30–103 for Pavia University dataset, and 196–110–60–30–196 for Salinas dataset. LPNPE [38] minimizes the distance of the spatial local pixel neighborhood. SSRLDE [38] preserves not only the spectral-domain local Euclidean neighborhood class relations but also the spatial-domain local pixel neighborhood structures. SSMRPE [39] shares the same DR concept as LLE. SSLDP [40] designs a weighted within neighborhood scatter to reveal the similarity of spatial neighbors. Among them, SSRLDE [38] and SSLDP [40] are supervised and require class labels to implement dimensionality reduction, while others are unsupervised.

For the fairness of the experimental comparison, the numbers of the nearest neighbor samples k of LE and LLE are the same as that of DFCEN_LE and DFCEN_LLE in the following experiments. We also choose the optimal parameters of their source literature for LPNPE [38], SSRLDE [38], SSMRPE [39], SSLDP [40]. In all the experiments below, all algorithms including DFCEN use raw data (that is not filtered to de-noise and smoothen pixels). For this reason, the results of the comparative experiments in this paper are different from those in the source literature (they usually use de-noising and smooth pixels).

Moreover, two classifiers support vector machines (SVM) and k nearest neighbor (KNN) are employed for classifying dimensionality reduction results. In fact, the number of the nearest neighbor of KNN is equal to 1. In all experiments, we randomly divide each HSI dataset into training and test sets. It should be emphasized that the training set is used to train the dimensionality reduction models and classifiers for supervised algorithms while that is only used to train classifiers for unsupervised algorithms. Actually, all samples in a HSI dataset are utilized to train the dimensionality reduction models for unsupervised methods. Overall classification accuracy (OA), average classification accuracy (AA), and the kappa coefficient κ are used to evaluate classification performance. To robustly evaluate the results with different dimensionality reduction algorithms, we repeat 10 times for each experiment.

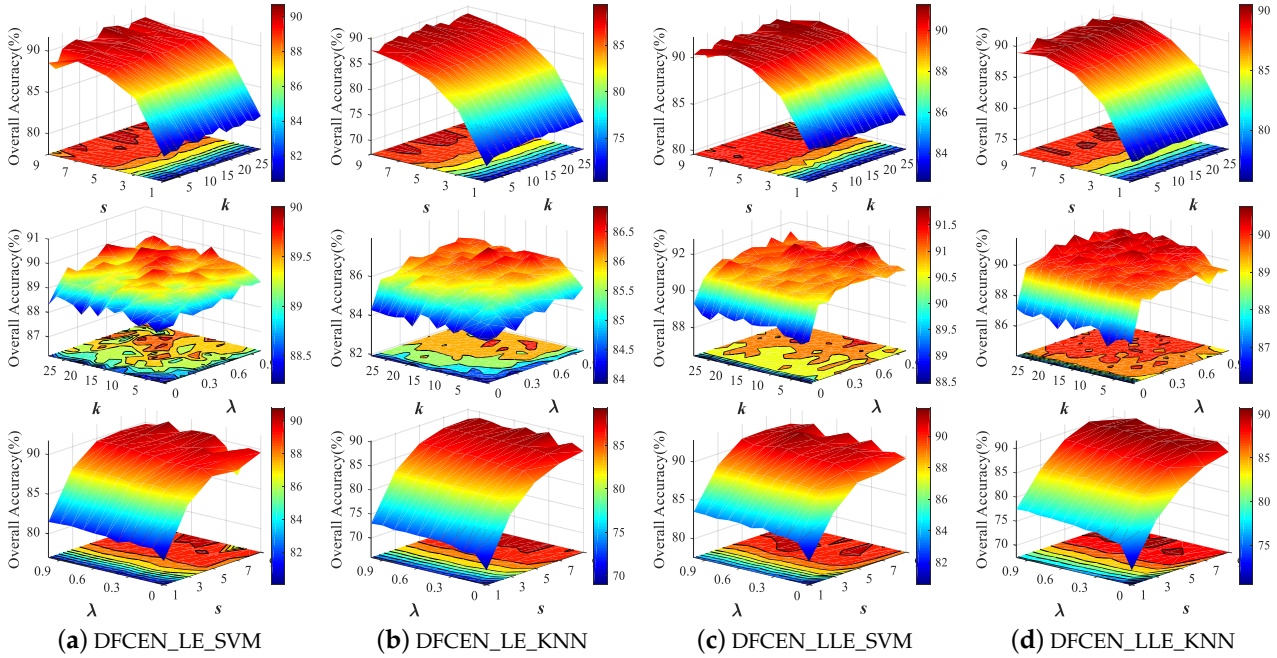
4.3. Parameters Analysis

Both DFCEN_LE and DFCEN_LLE have three parameters that need to be set manually, including nearest neighbor number k, spatial window size s and trade-off parameter λ. In order to analyze the influence of three parameters on dimensionality reduction, we conduct parameter tuning experiments on three HSI datasets. 10% in each class are randomly selected as the training set and the remaining samples are the testing set for two classifiers. Figure 10 shows the classification accuracy from DFCEN with different parameters on Indian Pines dataset, where the parameter range is set to: $k = \{1, 3, ..., 29\}$, $s = \{1, 3, ..., 9\}$, $\lambda = \{0, 0.1, 0.2, ..., 1\}$ and the fixed values are set to $k = 19$, $s = 5$, $\lambda = 0.4$ to analyze the other two parameters.

From Figure 10, the effects of the three parameters on DFCEN_LE and DFCEN_LLE are almost the same. The classification accuracy increases significantly with the increase of s when k or λ is fixed, which means that spatial information is important for DR. But the classification accuracy tends to decline when s continues to increase, because the large spatial window may contain heterogeneous samples which interfere with the extraction of spatial homogeneous information. Meanwhile, the classification accuracy increases with the increase of λ and k when s is fixed. In particular, the change of λ from zero has led to a significant improvement in classification, which proves that the specific learning task (this is embodied in the embedding term) of exploring and preserving the relationships among samples can effectively enhance the discriminability and separability of low-dimensional features and the proposed DFCEN is meaningful. Through a simple parameter tuning experiment, the three parameters of DFCEN_LE and DFCEN_LLE on the three datasets are set as shown in Table 2.

Table 2. Three parameter Settings for DFCEN_LE and DFCEN_LLE on three datasets.

	DFCEN_LLE			**DFCEN_LE**		
Parameters	λ	s	k	λ	s	k
Indian Pines	0.5	5	20	0.3	5	15
Pavia U	0.5	5	90	1	5	400
Salinas	0.3	7	120	0.5	7	600



(**a**) DFCEN_LE_SVM (**b**) DFCEN_LE_KNN (**c**) DFCEN_LLE_SVM (**d**) DFCEN_LLE_KNN

Figure 10. Classification overall accuracy with respect to different parameters of deep fully convolutional embedding network (DFCEN) on Indian Pines dataset from two classifiers.

4.4. Convergence and Discriminant Analysis

To illustrate the convergence of DFCEN, the learning curves of the embedding and reconstruction terms of DFCEN_LLE and DFCEN_LE on three datasets are present in Figure 11, in which the parameters have been initialized by CAEs. The *x*-axis represents the number of learning parameter updates that are performed after learning each batch of samples (a batch contains 50 samples). The curve represents the error values of two terms in objective function after one iteration (namely, all samples have been learned). (a)–(c) and (g)–(i) is about DFCEN_LLE, where two terms on three datasets all can remain convergent and obtain small error values after repeated iterations. (d)–(f) and (j)–(l) is about DFCEN_LE. From that, the error values of two terms on the Indian Pines and Salinas datasets can remain consistently convergent as the number of iterations increases. However, the error values of the reconstruction term in the early learning stage on the Pavia University dataset does not converge but increases. The reason is probably high trade-off parameter λ ($\lambda = 1$ shown in Table 2) and overfitting occurred in pretraining where the objective function of CAEs is consistent with the reconstruction term. Nevertheless, two terms on the Pavia University dataset eventually converge to a small error value as the number of iterations increases. Accordingly, DFCEN_LLE and DFCEN_LE can achieve a good convergence, from which the low-dimensional features not only preserves the original relationship among samples, but also retains the original intrinsic information in HSIs.

To analyze the discriminability and separability of the low-dimensional features from DFCEN, t-SNE is used to visualize the low-dimensional data of DFCEN comparing the raw data. The 2-dimensional features obtained by t-SNE on three datasets are shown in Figure 12 where different colors stand for different classes. Figure 12 shows all class samples for the Indian Pines and Pavia University datasets and randomly 80% for the Salinas dataset due to the large number of the class samples. As we can observe from these visualizations, the dimensionality reduction results from DFCEN are more discriminative than the raw HSIs data. Owing to DFCEN, the separability among different classes in the low-dimensional space is significantly improved compared to the original space. The reason is that DFCEN not only maintains the original intrinsic information but also preserves the original relationship among samples. In particular, DFCEN_LLE preserves

the original reconstructed relationship between each sample and its k nearest neighbors, while DFCEN_LE keeps each sample as close as possible to its k nearest neighbors, since there is a high probability that each sample and its neighbor belong to the same class. As a result, from Figure 12, the same classes from DFCEN are clustered together and the different classes are effectively separated.

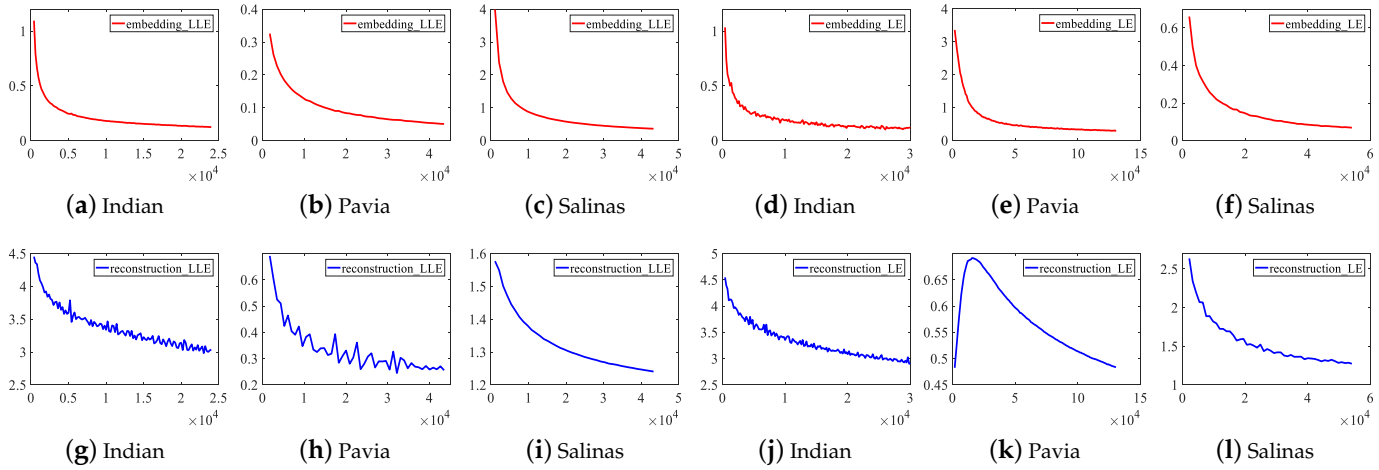


Figure 11. The learning curves of the embedding and reconstruction terms of DFCEN_LLE and DFCEN_LE on three datasets.

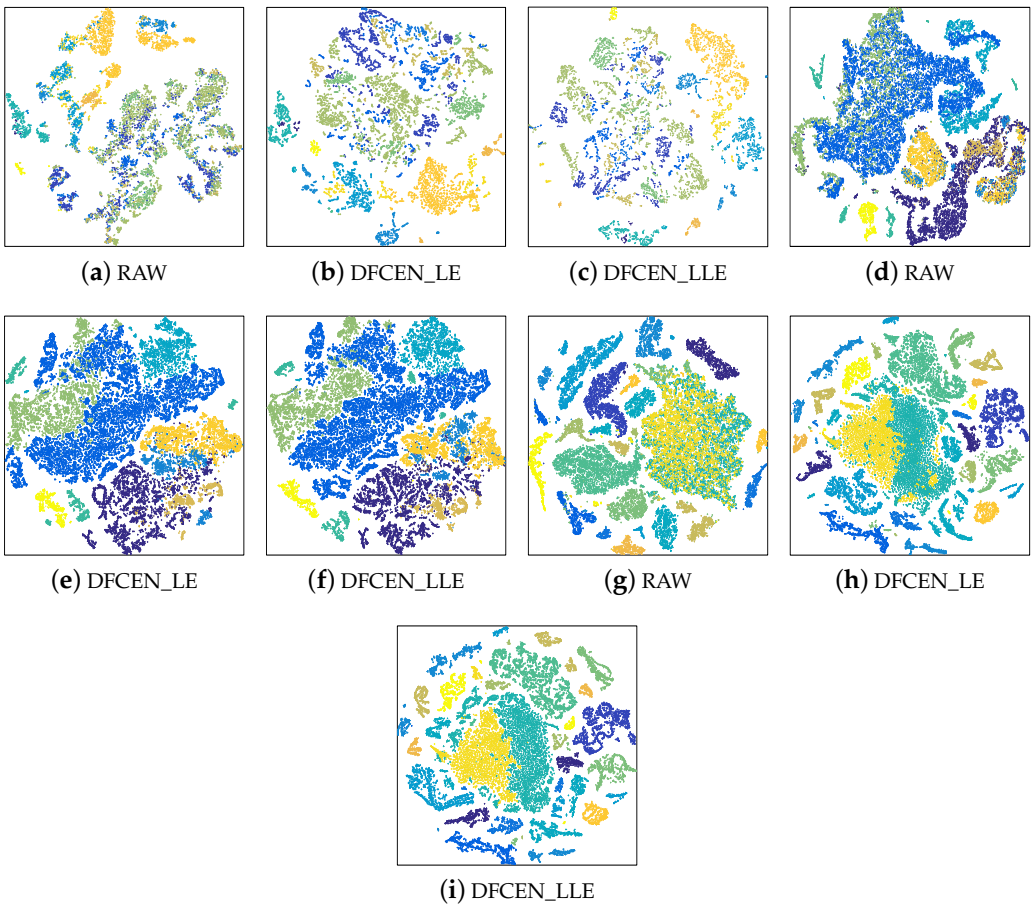


Figure 12. The two-dimensional features obtained by t-SNE from the raw data and the low-dimensional features of DFCEN on three datasets: (**a**–**c**) Indian Pines, (**d**–**f**) Pavia University, (**g**–**i**) Salinas.

4.5. Classification Performance

In this subsection, we examine the classification performance of dimensionality reduction results on three datasets. SVM and KNN are used to classify dimensionality reduction results to reduce the influence of classifiers. Firstly, in order to analyze the classification performance under different classification conditions, we randomly selected 5%, 10% and 15% of samples from each class as training set, and other samples are tested. The training set and test set are applied to all algorithms in the manner described in Section 4.2.

Table 3 shows the overall classification accuracy of the dimensionality reduction results (dim = 30) from different algorithms on three datasets, where the OA values is the average of 10 experiments under the same classification conditions. From Table 3, we can see that the classification OA values of all dimensionality reduction algorithms improve as the proportion of training samples increases since more training data can provide more class information for classifiers and supervised dimensionality reduction algorithms. The highest OA value under the same classification condition has been marked in bold.

As we have seen, the spatial–spectral combined algorithm, LPNPE [38], SSRLDE [38], SSMRPE [39], SSLDP [40] and DFCEN, are superior to the spectral-based algorithm, LE, LLE and SAE, which indicates that spatial features are beneficial to the dimensionality reduction of HSIs. Neural network based methods, SAE and DFCEN, are superior to traditional dimensionality reduction algorithms, which testifies that neural network is suitable for dimensionality reduction of HSIs. Compared with other algorithms in this paper, the dimensionality reduction results of DFCEN has the best classification performance for three datasets under two classifiers. In particular, DFCEN achieves superior classification accuracy even with only 5% of the training samples of classifiers.

Table 3. Classification accuracy of dimensionality reduction results (dim = 30) of different algorithms using SVM and KNN classifiers with different proportions of training samples on three datasets.

Dataset			RAW	LE	LLE	SAE	LPNPE	SSRLDE	SSMRPE	SSLDP	DFCEN_LE	DFCEN_LLE
Indian	5%	SVM	75.40	74.68	59.29	77.81	82.47	78.26	73.78	70.58	84.55	**86.30**
		KNN	65.00	73.48	56.08	66.59	81.71	80.45	74.97	70.41	81.01	**81.85**
	10%	SVM	76.06	77.99	61.26	81.64	86.24	83.11	78.93	74.07	90.25	**91.18**
		KNN	68.70	77.48	57.85	70.52	85.39	81.56	77.66	72.45	86.74	**88.52**
	15%	SVM	83.90	78.88	62.03	83.58	88.13	85.81	81.48	75.01	92.60	**93.28**
		KNN	70.63	79.04	58.91	72.25	87.11	82.62	79.66	72.96	89.89	**91.81**
Pavia U	5%	SVM	93.56	80.20	89.33	92.72	89.63	89.10	88.05	78.55	96.09	**96.32**
		KNN	84.96	73.95	81.22	82.88	91.69	90.35	86.05	77.74	**94.00**	93.45
	10%	SVM	94.49	81.09	90.40	93.49	91.13	90.57	89.70	80.03	**97.25**	97.05
		KNN	86.63	74.60	82.49	84.02	92.41	91.43	87.11	77.95	**95.73**	95.37
	15%	SVM	94.82	81.31	90.99	93.80	92.11	91.37	90.69	80.81	**97.76**	97.57
		KNN	87.37	74.84	83.41	84.72	92.84	92.01	87.66	78.86	**96.39**	96.26
Salinas	5%	SVM	93.37	85.85	90.14	92.32	92.82	91.82	93.51	92.54	96.12	**96.87**
		KNN	86.93	81.95	86.01	88.15	94.13	91.74	90.96	93.58	95.53	**97.11**
	10%	SVM	94.04	86.23	90.82	93	93.88	93.23	94.12	93.01	96.82	**97.64**
		KNN	88.13	82.84	86.76	89.18	94.51	92.14	91.85	93.96	96.84	**98.29**
	15%	SVM	94.58	86.48	91.11	93.2	94.42	93.94	94.46	93.24	97.09	**98.02**
		KNN	88.66	83.36	87.37	89.74	94.85	92.41	92.00	94.22	97.53	**98.69**

Secondly, in order to analyze the classification performances per class of different algorithms, 10% of samples per class are randomly selected as training samples and others are as test samples. The individual class classification accuracy, OA, AA, and κ on three datasets are shown in Tables 4–6. The highest value of each item has been marked in bold. Figures 13–15 show the corresponding classification maps of different algorithms on three datasets. From Tables 4–6, the supervised algorithms, SSRLDE and SSLDP, give unsatisfactory classification results in Indian Pines and Pavia University datasets due to the absence of pixel filtering, which indicates that SSRLDE and SSLDP are very sensitive to noise pixels. Meanwhile, as two unsupervised algorithms, DFCEN_LLE and DFCEN_LE achieved the highest classification accuracy in most classes, even with OA, AA and κ achieving the best. Especially for class 9 in Indian Pines, class 3 in Pavia and class 15 in Salinas, DFCEN obtain high classification accuracy while other algorithms are poor because

of the difficulty in classifying these classes. In terms of OA, DFCEN is approximately 4% better than that of the second best algorithm.

Table 4. Classification accuracy of each class (DIM = 30) for Indian Pines datasets via SVM and KNN classifiers.

Class		RAW	LE	LLE	SAE	LPNPE	SSRLDE	SSMRPE	SSLDP	DFCEN_LE	DFCEN_LLE
C1	SVM	19.5	19.5	34.1	75.6	68.3	**90.0**	58.5	62.5	56.1	85.4
	KNN	43.9	48.8	22.0	29.3	**80.5**	70.0	53.7	62.5	46.3	73.2
C2	SVM	77.7	72.8	40.5	80.4	86.4	78.1	74.6	76.0	**89.6**	85.9
	KNN	56.0	70.6	43.3	66.5	80.2	75.2	69.3	69.9	81.2	**83.1**
C3	SVM	68.3	43.8	9.9	67.3	80.1	69.3	62.8	63.5	**90.2**	87.3
	KNN	53.7	62.1	33.1	52.3	76.0	67.2	62.5	53.7	76.2	**79.1**
C4	SVM	56.8	21.1	24.9	62.0	73.2	**79.3**	57.7	45.5	77.9	78.9
	KNN	41.3	30.0	35.2	41.8	**73.7**	62.0	64.3	54.9	52.6	67.6
C5	SVM	90.3	77.2	73.1	88.7	93.8	94.3	89.7	86.4	97.0	**98.2**
	KNN	79.1	81.6	72.4	77.2	94.9	90.8	90.6	79.5	94.9	**96.1**
C6	SVM	93.6	98.0	95.9	93.6	98.2	93.9	94.5	94.1	96.8	**99.1**
	KNN	93.8	91.5	80.8	93.3	97.9	96.5	94.5	89.8	**98.9**	98.3
C7	SVM	88.0	92.0	68.0	64.0	**100**	90.9	84.0	72.7	88.0	96.0
	KNN	88.0	92.0	44.0	80.0	92.0	81.8	92.0	86.4	88.0	**100**
C8	SVM	97.9	97.7	90.2	98.4	**99.8**	98.6	99.3	99.3	98.1	99.3
	KNN	94.0	93.0	89.5	95.1	**100**	99.3	97.0	99.8	99.3	**100**
C9	SVM	5.6	11.1	0	50.0	88.9	85.7	44.4	21.4	**100**	83.3
	KNN	16.7	22.2	44.4	33.3	66.7	**100**	38.9	35.7	**100**	**100**
C10	SVM	71.3	72.2	25.5	71.5	81.9	74.9	74.6	58.7	85.4	**91.0**
	KNN	61.6	74.6	40.8	58.9	81.8	78.5	73.6	58.1	90.1	**92.1**
C11	SVM	83.9	86.3	87.5	85.8	83.0	82.1	78.1	78.2	87.5	**89.7**
	KNN	71.5	81.8	60.3	72.7	85.8	86.9	79.6	83.8	**88.0**	87.9
C12	SVM	71.9	56.6	21.5	70.6	83.5	81.1	59.9	67.8	86.3	**88.8**
	KNN	40.8	50.6	27.2	57.7	**85.2**	70.0	63.9	69.9	70.4	73.2
C13	SVM	93.5	85.3	90.2	96.7	99.5	95.1	96.7	96.7	99.5	**100**
	KNN	95.1	96.2	85.3	88.0	**98.9**	96.2	96.2	97.8	**98.9**	**98.9**
C14	SVM	95.7	94.5	94.4	89.6	95.0	92.9	93.3	96.1	**96.9**	96.1
	KNN	85.4	92.0	90.9	86.9	94.3	91.3	90.9	95.3	95.9	**97.1**
C15	SVM	61.1	78.4	15.9	53.3	78.1	69.2	56.8	43.8	**83.0**	**83.0**
	KNN	36.6	69.7	33.7	34.0	74.1	67.1	55.9	36.0	70.9	**83.6**
C16	SVM	83.3	**97.6**	85.7	81.0	86.9	91.7	85.7	84.5	86.9	**97.6**
	KNN	85.7	**98.8**	88.1	83.3	88.1	92.9	90.5	84.5	94.0	92.9
OA	SVM	80.5	77.7	61.3	81.3	87.0	83.1	78.6	77.2	90.3	**91.1**
	KNN	67.5	77.2	58.0	70.5	86.3	82.7	78.1	76.2	86.5	**88.5**
AA	SVM	72.4	69.0	53.6	76.8	87.3	85.4	75.7	71.7	88.7	**91.2**
	KNN	65.2	72.2	55.7	65.7	85.6	82.9	75.8	72.3	84.1	**88.9**
κ	SVM	78.5	74.4	54.3	78.6	85.2	80.8	75.5	73.8	88.9	**89.9**
	KNN	63.7	74.0	52.0	66.3	84.4	80.3	75.0	72.5	84.6	**86.9**

Table 5. Classification accuracy of each class (DIM = 30) for Pavia University datasets via SVM and KNN classifiers.

Class		RAW	LE	LLE	SAE	LPNPE	SSRLDE	SSMRPE	SSLDP	DFCEN_LE	DFCEN_LLE
C1	SVM	94.9	84.8	90.7	93.9	91.5	90.1	91.3	79.5	**97.5**	97.5
	KNN	87.4	80.1	80.2	85.4	91.7	89.5	84.5	78.4	93.9	**94.5**
C2	SVM	98.4	97.0	97.2	97.6	96.9	96.1	96.4	93.1	**99.2**	99.0
	KNN	94.4	83.4	94.7	92.1	97.4	97.7	95.9	94.5	**99.6**	**99.6**
C3	SVM	80.7	31.1	71.7	75.9	71.6	71.9	70.2	56.4	**92.3**	90.8
	KNN	65.2	40.4	56.5	60.5	78.3	78.1	63.5	59.4	**87.8**	86.7
C4	SVM	95.3	77.9	91.1	93.1	92.2	92.8	90.4	69.1	**98.5**	97.1
	KNN	84.0	74.8	74.7	84.0	92.5	87.2	85.6	64.9	**92.7**	92.5
C5	SVM	99.7	98.6	99.8	99.3	99.8	99.9	99.8	99.8	**100**	**100**
	KNN	98.8	99.1	99.5	99.5	99.6	99.8	99.8	99.8	**100**	99.9
C6	SVM	87.3	31.2	77.3	84.0	85.6	83.0	81.0	73.2	93.0	**93.6**
	KNN	66.1	46.0	63.7	59.7	**88.0**	80.7	77.6	72.6	87.7	86.3
C7	SVM	87.5	70.7	72.5	82.6	72.3	71.4	67.3	44.9	**90.6**	87.5
	KNN	81.5	58.4	69.4	80.7	88.6	88.1	74.7	51.5	**94.0**	92.3
C8	SVM	88.1	86.0	87.2	89.4	79.2	79.4	77.5	55.8	94.0	**96.1**
	KNN	81.9	68.2	71.6	81.9	80.2	85.1	72.5	54.7	**93.5**	92.4

Table 5. *Cont.*

Class		RAW	LE	LLE	SAE	LPNPE	SSRLDE	SSMRPE	SSLDP	DFCEN_LE	DFCEN_LLE
C9	SVM	**99.9**	99.6	99.6	99.8	99.8	99.8	**99.9**	74.9	**99.9**	**99.9**
	KNN	99.6	99.6	99.8	**100**	**100**	99.9	98.1	89.4	99.8	99.3
OA	SVM	94.2	81.1	90.7	93.0	91.0	90.2	89.8	80.2	**97.1**	97.0
	KNN	86.3	74.5	83.0	84.3	92.5	91.4	87.1	80.9	**95.6**	95.2
AA	SVM	92.4	75.2	87.5	90.6	87.7	87.2	86.0	71.8	**96.1**	95.7
	KNN	84.3	72.2	78.9	82.6	90.7	89.6	83.6	73.9	**94.3**	93.7
κ	SVM	92.4	74.0	87.5	90.6	88.0	87.0	86.3	73.5	**96.2**	96.1
	KNN	82.0	66.1	77.1	79.0	90.0	88.6	82.8	74.3	**94.1**	93.7

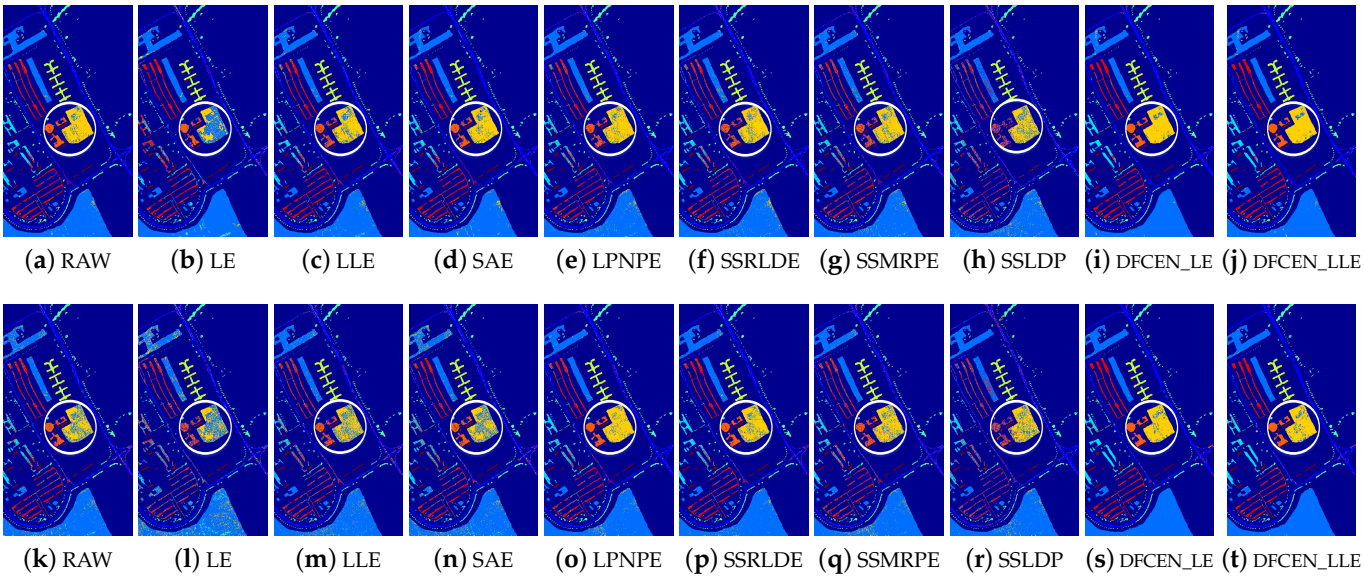
Table 6. Classification accuracy of each class (DIM = 30) for Salinas datasets via SVM and KNN classifiers.

Class		RAW	LE	LLE	SAE	LPNPE	SSRLDE	SSMRPE	SSLDP	DFCEN_LE	DFCEN_LLE
C1	SVM	99.8	97.5	98.6	99.0	99.9	99.4	99.4	99.9	**100**	**100**
	KNN	98.3	97.1	98.4	98.6	99.9	99.2	99.5	99.9	99.5	**100**
C2	SVM	99.9	98.8	99.2	99.8	99.9	99.8	99.9	99.9	**100**	**100**
	KNN	99.7	98.3	99.5	99.7	**100**	99.8	**100**	**100**	99.9	99.9
C3	SVM	**99.9**	96.9	97.8	99.6	99.7	99.2	99.7	99.8	99.7	99.7
	KNN	98.8	95.7	85.2	99.0	99.9	99.7	99.8	**100**	99.9	99.7
C4	SVM	99.4	98.6	99.4	99.5	97.4	98.7	99.8	99.2	**100**	**100**
	KNN	99.0	97.5	98.3	99.5	99.2	99.3	**99.9**	99.8	99.8	99.5
C5	SVM	99.2	96.6	98.7	98.2	98.7	99.3	99.2	98.8	**100**	99.9
	KNN	98.5	97.3	95.9	98.0	99.1	99.2	99.5	98.5	99.4	**100**
C6	SVM	99.8	99.5	**100**	99.8	99.9	**100**	**100**	99.9	**100**	**100**
	KNN	99.8	99.2	99.9	99.7	99.9	**100**	99.9	99.9	**100**	**100**
C7	SVM	99.8	99.3	99.8	99.9	99.9	99.8	**100**	99.9	99.9	**100**
	KNN	99.6	98.0	99.9	99.3	99.9	99.9	**100**	99.9	**100**	**100**
C8	SVM	90.3	81.1	86.2	89.7	90.9	87.5	88.4	88.7	93.1	**95.0**
	KNN	75.1	66.3	72.0	76.2	87.7	82.1	80.9	88.7	92.6	**94.3**
C9	SVM	99.9	98.6	99.8	**100**	99.6	99.1	99.7	**100**	99.9	99.8
	KNN	99.4	98.3	99.2	99.5	99.9	99.8	99.9	**100**	99.9	99.8
C10	SVM	96.9	86.8	90.7	94.7	98.3	97.7	98.8	97.9	**99.4**	99.3
	KNN	90.6	81.6	89.6	90.9	98.3	97.1	98.0	97.6	98.5	**99.2**
C11	SVM	98.9	87.2	96.1	96.0	98.9	98.4	99.7	99.4	98.1	**99.9**
	KNN	94.9	87.3	91.1	97.5	97.8	99.6	99.8	99.5	**100**	**100**
C12	SVM	99.3	98.1	99.2	99.9	99.6	98.6	99.9	**100**	**100**	**100**
	KNN	99.3	95.2	97.1	99.9	**100**	99.9	**100**	**100**	**100**	**100**
C13	SVM	97.9	97.5	98.4	99.0	95.8	98.7	99.6	99.5	**100**	**100**
	KNN	97.6	96.1	97.5	96.1	99.2	98.8	99.0	99.5	**100**	**100**
C14	SVM	97.0	91.4	92.7	95.1	96.1	96.4	97.6	97.0	99.7	**99.9**
	KNN	93.8	91.3	94.1	95.6	98.2	96.7	98.4	96.7	**99.6**	99.3
C15	SVM	73.5	44.1	61.6	63.9	73.2	74.8	76.4	67.3	85.3	**91.5**
	KNN	60.5	47.4	60.3	64.1	81.4	73.0	68.6	77.8	87.7	**95.3**
C16	SVM	98.8	92.7	99.2	98.5	99.0	99.0	99.6	98.6	98.8	**100**
	KNN	98.2	91.5	99.0	96.9	99.8	99.5	99.5	98.6	99.4	**99.9**
OA	SVM	93.7	86.2	90.8	92.2	94.0	93.4	94.1	92.9	96.5	**97.7**
	KNN	87.9	82.9	86.7	89.0	94.7	92.2	91.6	94.3	96.6	**98.1**
AA	SVM	96.9	91.5	94.8	95.8	96.7	96.6	97.4	96.6	98.4	**99.1**
	KNN	93.9	89.9	92.3	94.4	97.5	96.5	96.4	97.3	98.5	**99.2**
κ	SVM	93.3	84.6	89.7	91.4	93.3	92.6	93.5	92.1	96.1	**97.5**
	KNN	86.9	80.9	85.2	87.8	94.0	91.3	90.6	93.6	96.2	**97.8**

Figures 13–15 visually show the classification maps of the DR results (DIM = 30) of different algorithms. From that, it can be observed that DFCEN has significant regional classification uniformity because DFCEN not only guarantees the intrinsic information of HSIs but also explores and maintains the relationship among samples and their nearest neighbors. Especially for classes 3, 9, 10, 12, and 15 of the Indian Pines dataset, classes 6 and 7 of the Pavia University dataset, and classes 8 and 15 of the Salinas dataset (these classes have been circled in white in Figures 13–15, DFCEN performs much better than the other methods under two classifiers.

Thirdly, to analyze the influence of different dimensions on each algorithm, Figure 16 shows the changes of OA with two classifiers on three datasets when the dimensionality ranges from 5 to 50 with the step length of 5. From that, the OAs of most algorithms improve with the increase of dimensions and tend to be stable when the dimension increases to a certain degree. The reason is that the higher the feature dimension is, the more information it can provide for the classification, but it will reach saturation when the feature dimension continues to increase. Moreover, in Figure 16, spatial–spectral DR methods, LPNPE, SSRLDE, SSMRPE, SSRLDE and DFCEN, are generally superior to spectral-based methods, LE, LLE and SAE. In particular, DFCEN achieves almost the best classification on the results of different dimensions compared with other algorithms.

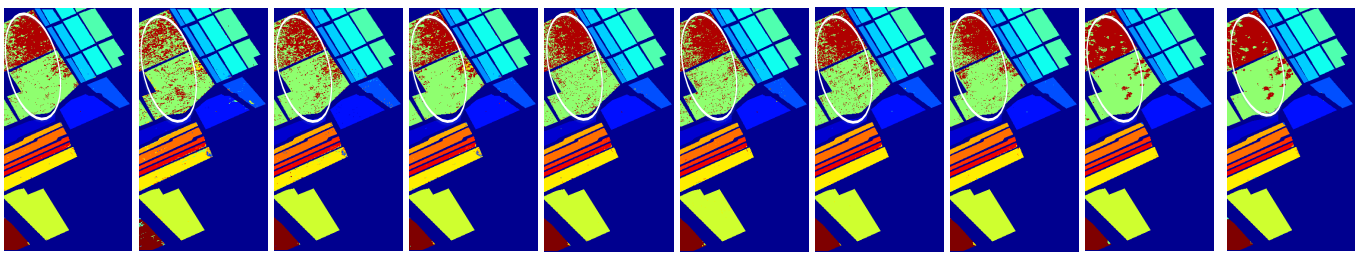
Figure 16 also shows that the classification OAs of LE and LLE are relatively poor. However, DFCEN_LE and DFCEN_LLE have satisfactory classification performance when the concepts of LE and LLE are introduced to DFCEN. The reason may be summarized as follows: (1) the fully convolutional network of DFCEN can effectively obtain the spatial–spectral information of HSIs by layer-by-layer feature extraction, (2) the reconstruction term, as a regularization term corresponding to the embedding term, can constrain low-dimensional features to retain the intrinsic information.



(**a**) RAW (**b**) LE (**c**) LLE (**d**) SAE (**e**) LPNPE (**f**) SSRLDE (**g**) SSMRPE (**h**) SSLDP (**i**) DFCEN_LE (**j**) DFCEN_LLE

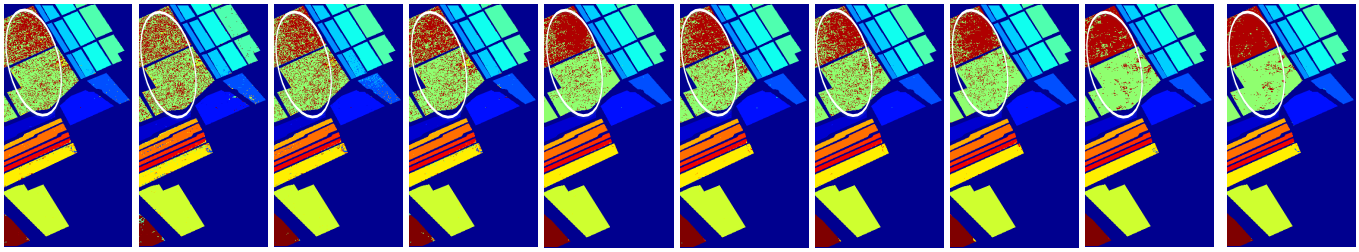
(**k**) RAW (**l**) LE (**m**) LLE (**n**) SAE (**o**) LPNPE (**p**) SSRLDE (**q**) SSMRPE (**r**) SSLDP (**s**) DFCEN_LE (**t**) DFCEN_LLE

Figure 13. Classification maps with two classifiers of different methods on the University of Pavia dataset (dim = 30). (**a**–**j**) are for KNN and (**k**–**t**) are for SVM. (**i**,**j**) and (**s**,**t**) are the classification result of DFCEN.



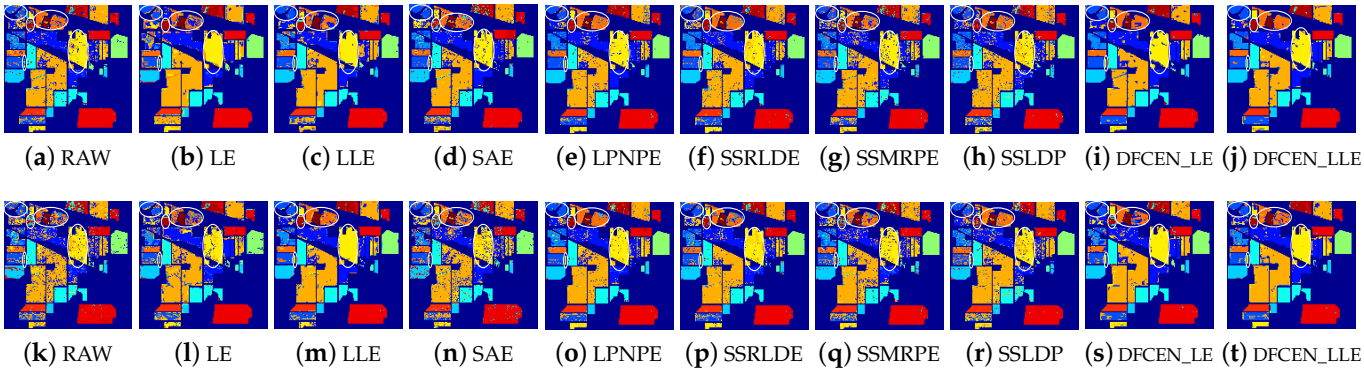
(**a**) RAW (**b**) LE (**c**) LLE (**d**) SAE (**e**) LPNPE (**f**) SSRLDE (**g**) SSMRPE (**h**) SSLDP (**i**) DFCEN_LE (**j**) DFCEN_LLE

Figure 14. *Cont.*



(**k**) RAW (**l**) LE (**m**) LLE (**n**) SAE (**o**) LPNPE (**p**) SSRLDE (**q**) SSMRPE (**r**) SSLDP (**s**) DFCEN_LE (**t**) DFCEN_LLE

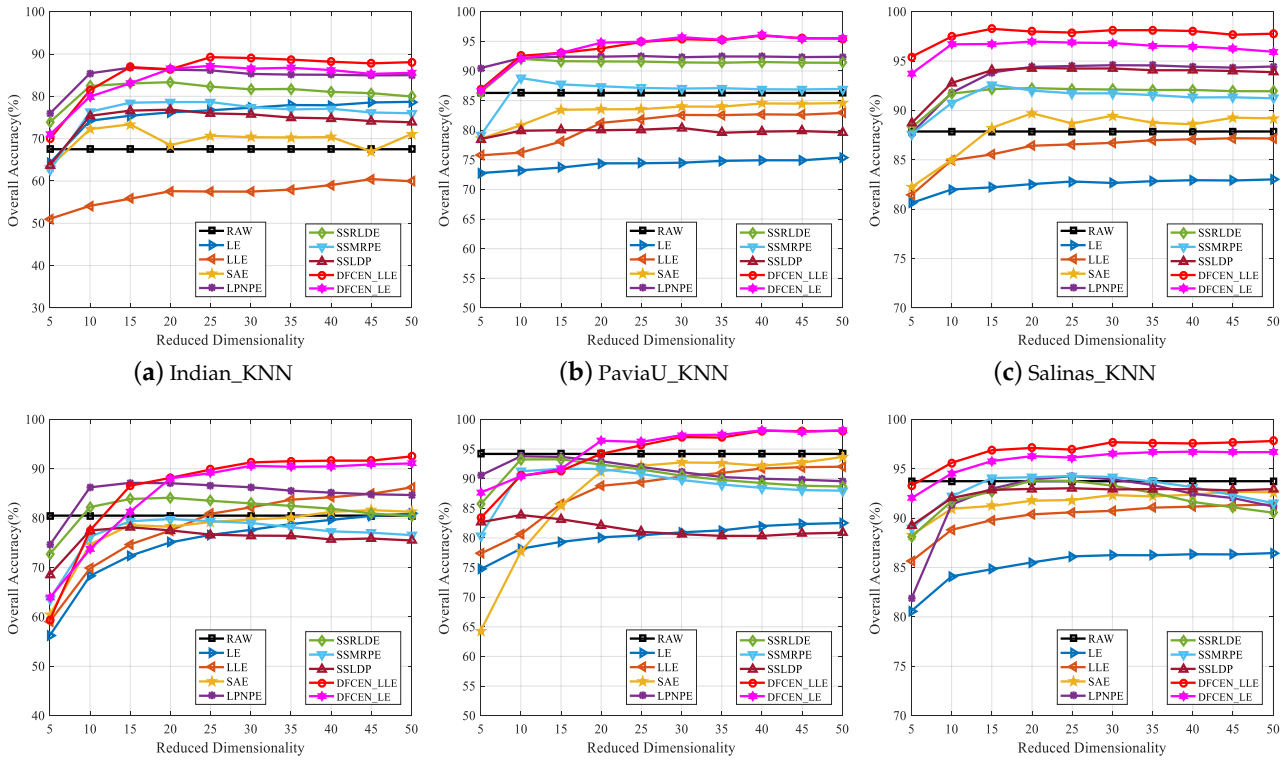
Figure 14. Classification maps with two classifier of different methods on Salinas data set (dim = 30). (**a**–**j**) are for KNN and (**k**–**t**) are for SVM. (**i**,**j**) and (**s**,**t**) are the classification result of the proposed DFCEN.



(**a**) RAW (**b**) LE (**c**) LLE (**d**) SAE (**e**) LPNPE (**f**) SSRLDE (**g**) SSMRPE (**h**) SSLDP (**i**) DFCEN_LE (**j**) DFCEN_LLE

(**k**) RAW (**l**) LE (**m**) LLE (**n**) SAE (**o**) LPNPE (**p**) SSRLDE (**q**) SSMRPE (**r**) SSLDP (**s**) DFCEN_LE (**t**) DFCEN_LLE

Figure 15. Classification maps of different methods on Indian Pines dataset (dim = 30) via two classifiers. (**a**–**j**) are for *k* nearest neighbor (KNN) and (**k**–**t**) are for support vector machines (SVM). (**i**,**j**) and (**s**,**t**) are the classification result of DFCEN.



(**a**) Indian_KNN (**b**) PaviaU_KNN (**c**) Salinas_KNN

(**d**) Indian_SVM (**e**) PaviaU_SVM (**f**) Salinas_SVM

Figure 16. Classification overall accuracy of reduced dimensionality (DIM = 5∼50) on three datasets with SVM and KNN classifiers.

5. Conclusions

In this paper, a novel unsupervised DFCEN was proposed for HSIs dimensionality reduction. Different from the existing unsupervised CNN-based method which only focuses on data reconstruction, DFCEN was designed to not only ensure data reconstruction but also realize the learning of specific tasks. In DFCEN, convolutional subnetwork is for dimensionality reduction and specific task learning while deconvolutional subnetwork is for data reconstruction. A novel objective function was proposed, including two terms: embedding term of the specific task and reconstruction term of data reconstruction. The former enhance the discriminant ability of low-dimensional features and the latter maintain the original intrinsic information. In this paper, exploring and maintaining relationships between samples as a specific task to improve dimensionality reduction performance, while the dimensionality reduction concepts of LLE and LE are introduced into DFCEN. Experimental results on three hyperspectral datasets prove the superior classification performance of the dimensionality reduction results from DFCEN_LLE and DFCEN_LE.

In our future work, different dimensionality reduction concepts and objective functions designed according to specific requirements will be applied to DFCEN to achieve DR and the idea of the combination of LE and LLE will be tried. In addition, we will try to apply DFCEN to other areas.

Author Contributions: Conceptualization, N.L. and M.Z.; methodology, N.L.; software, N.L.; validation, N.L., M.Z. and T.W.; formal analysis, N.L.; investigation, N.L.; resources, D.Z.; data curation, J.S.; writing—original draft preparation, N.L.; writing—review and editing, N.L.; visualization, N.L.; supervision, M.G.; project administration, D.Z.; funding acquisition, J.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by National Natural Science Foundation of China (Grant No. 62076204), the National Natural Science Foundation of Shaanxi Province under Grantnos. 2018JQ6003 and 2018JQ6030, the China Postdoctoral Science Foundation (Grant nos. 2017M613204 and 2017M623246).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Murphy, R.J.; Monteiro, S.T.; Schneider, S. Evaluating Classification Techniques for Mapping Vertical Geology Using Field-Based Hyperspectral Sensors. *IEEE Trans. Geosci. Remote Sens.* **2012**, *50*, 3066–3080. [CrossRef]
2. Ryan, J.P.; Davis, C.O.; Tufillaro, N.B.; Kudela, R.M.; Gao, B.C. Application of the hyperspectral imager for the coastal ocean to phytoplankton ecology studies in Monterey Bay, CA, USA. *Remote Sens.* **2014**, *6*, 1007–1025. [CrossRef]
3. Pi, W.; Du, J.; Liu, H.; Zhu, X. Desertification Glassland Classification and Three-Dimensional Convolution Neural Network Model for Identifying Desert Grassland Landforms with Unmanned Aerial Vehicle Hyperspectral Remote Sensing Images. *J. Appl. Spectrosc.* **2020**, *87*, 309–318. [CrossRef]
4. Ofner, J.; Kamilli, K.A.; Eitenberger, E.; Friedbacher, G.; Lendl, B.; Held, A.; Lohninger, H. Chemometric analysis of multisensor hyperspectral images of precipitated atmospheric particulate matter. *Anal. Chem.* **2015**, *87*, 9413–9420. [CrossRef] [PubMed]
5. de la Ossa, M.Á.F.; Amigo, J.M.; García-Ruiz, C. Detection of residues from explosive manipulation by near infrared hyperspectral imaging: A promising forensic tool. *Forensic Sci. Int.* **2014**, *242*, 228–235. [CrossRef]
6. Guo, X.; Huang, X.; Zhang, L.; Zhang, L. Hyperspectral image noise reduction based on rank-1 tensor decomposition. *ISPRS J. Photogramm. Remote Sens.* **2013**, *83*, 50–63. [CrossRef]
7. Jia, X.; Kuo, B.C.; Crawford, M.M. Feature Mining for Hyperspectral Image Classification. *Proc. IEEE* **2013**, *101*, 676–697. [CrossRef]
8. Guo, B.; Gunn, S.R.; Damper, R.I.; Nelson, J.D.B. Band Selection for Hyperspectral Image Classification Using Mutual Information. *IEEE Geosci. Remote Sens. Lett.* **2006**, *3*, 522–526. [CrossRef]
9. Hughes, G. On the mean accuracy of statistical pattern recognizers. *IEEE Trans. Inf. Theory* **1968**, *14*, 55–63. [CrossRef]
10. Li, Y.; Qu, J.; Dong, W.; Zheng, Y. Hyperspectral pansharpening via improved PCA approach and optimal weighted fusion strategy. *Neurocomputing* **2018**, *315*, 371–380. [CrossRef]
11. Belkin, M.; Niyogi, P. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput.* **2003**, *15*, 1373–1396. [CrossRef]
12. Li, W.; Zhang, L.; Zhang, L.; Du, B. GPU parallel implementation of isometric mapping for hyperspectral classification. *IEEE Geosci. Remote Sens. Lett.* **2017**, *14*, 1532–1536. [CrossRef]

13. Bandos, T.V.; Bruzzone, L.; Camps-Valls, G. Classification of Hyperspectral Images With Regularized Linear Discriminant Analysis. *IEEE Trans. Geosci. Remote Sens.* **2009**, *47*, 862–873. [CrossRef]
14. He, X.; Niyogi, P. Locality preserving projections. In *Advances in Neural Information Processing Systems*; The MIT Press: Cambridge, MA, USA, 2004; pp. 153–160.
15. He, X.; Cai, D.; Yan, S.; Zhang, H.J. Neighborhood preserving embedding. In Proceedings of the Tenth IEEE ICCV, Beijing, China, 17–20 October 2005; Volume 2, pp. 1208–1213.
16. Han, T.; Goodenough, D.G. Investigation of Nonlinearity in Hyperspectral Imagery Using Surrogate Data Methods. *IEEE Trans. Geosci. Remote Sens.* **2008**, *46*, 2840–2847. [CrossRef]
17. Bengio, Y.; Courville, A.C.; Vincent, P. Unsupervised feature learning and deep learning: A review and new perspectives. *CoRR* **2012**, *1*, 2012.
18. Paoletti, M.E.; Haut, J.M.; Plaza, J.; Plaza, A. Deep learning classifiers for hyperspectral imaging: A review. *ISPRS J. Photogramm. Remote Sens.* **2019**, *158*, 279–317. [CrossRef]
19. Paoletti, M.; Haut, J.; Plaza, J.; Plaza, A. A new deep convolutional neural network for fast hyperspectral image classification. *ISPRS J. Photogramm. Remote Sens.* **2018**, *145*, 120–147. [CrossRef]
20. Zhong, Z.; Li, J.; Luo, Z.; Chapman, M. Spectral–spatial residual network for hyperspectral image classification: A 3-D deep learning framework. *IEEE Trans. Geosci. Remote Sens.* **2017**, *56*, 847–858.
21. Han, M.; Cong, R.; Li, X.; Fu, H.; Lei, J. Joint spatial-spectral hyperspectral image classification based on convolutional neural network. *Pattern Recognit. Lett.* **2020**, *130*, 38–45. [CrossRef]
22. Mou, L.; Ghamisi, P.; Zhu, X.X. Unsupervised spectral–spatial feature learning via deep residual Conv–Deconv network for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **2017**, *56*, 391–406. [CrossRef]
23. Zhang, M.; Gong, M.; Mao, Y.; Li, J.; Wu, Y. Unsupervised feature extraction in hyperspectral images based on wasserstein generative adversarial network. *IEEE Trans. Geosci. Remote Sens.* **2018**, *57*, 2669–2688. [CrossRef]
24. Zhang, M.; Gong, M.; He, H.; Zhu, S. Symmetric All Convolutional Neural-Network-Based Unsupervised Feature Extraction for Hyperspectral Images Classification. *IEEE Trans. Cybern.* **2020**. [CrossRef]
25. Estévez, P.A.; Tesmer, M.; Perez, C.A.; Zurada, J.M. Normalized mutual information feature selection. *IEEE Trans. Neural Netw.* **2009**, *20*, 189–201. [CrossRef]
26. Chang, C.I.; Kuo, Y.M.; Chen, S.; Liang, C.C.; Ma, K.Y.; Hu, P.F. Self-Mutual Information-Based Band Selection for Hyperspectral Image Classification. *IEEE Trans. Geosci. Remote. Sens.* **2020**. [CrossRef]
27. Pan, Y.; Ge, S.S.; Al Mamun, A. Weighted locally linear embedding for dimension reduction. *Pattern Recognit.* **2009**, *42*, 798–811. [CrossRef]
28. Roweis, S.T.; Saul, L.K. Nonlinear dimensionality reduction by locally linear embedding. *Science* **2000**, *290*, 2323–2326. [CrossRef]
29. Wang, M.; Yu, J.; Niu, L.; Sun, W. Unsupervised feature extraction for hyperspectral images using combined low rank representation and locally linear embedding. In Proceedings of the 2017 IEEE ICASSP, New Orleans, LA, USA, 5–9 March 2017; pp. 1428–1431.
30. Li, B.; Li, Y.R.; Zhang, X.L. A survey on Laplacian eigenmaps based manifold learning methods. *Neurocomputing* **2019**, *335*, 336–351.
31. Ma, M.; Deng, T.; Wang, N.; Chen, Y. Semi-supervised rough fuzzy Laplacian Eigenmaps for dimensionality reduction. *Int. J. Mach. Learn. Cybern.* **2019**, *10*, 397–411. [CrossRef]
32. Seyfioğlu, M.S.; Özbayoğlu, A.M.; Gürbüz, S.Z. Deep convolutional autoencoder for radar-based classification of similar aided and unaided human activities. *IEEE Trans. Aerosp. Electron. Syst.* **2018**, *54*, 1709–1723. [CrossRef]
33. Azarang, A.; Manoochehri, H.E.; Kehtarnavaz, N. Convolutional autoencoder-based multispectral image fusion. *IEEE Access* **2019**, *7*, 35673–35683. [CrossRef]
34. Palsson, B.; Ulfarsson, M.O.; Sveinsson, J.R. Convolutional Autoencoder for Spectral–Spatial Hyperspectral Unmixing. *IEEE Trans. Geosci. Remote Sens.* **2020**, *59*, 535–549.
35. Fauvel, M.; Tarabalka, Y.; Benediktsson, J.A.; Chanussot, J.; Tilton, J.C. Advances in spectral-spatial classification of hyperspectral images. *Proc. IEEE* **2013**, *101*, 652–675. [CrossRef]
36. Plaza, A.; Martínez, P.; Pérez, R.; Plaza, J. Spatial/spectral endmember extraction by multidimensional morphological operations. *IEEE Trans. Geosci. Remote Sens.* **2002**, *40*, 2025–2041. [CrossRef]
37. Bouvrie, J. Notes on Convolutional Neural Networks. Available online: http://cogprints.org/5869/ (accessed on 13 February 2021).
38. Zhou, Y.; Peng, J.; Chen, C.P. Dimension reduction using spatial and spectral regularized local discriminant embedding for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **2014**, *53*, 1082–1095. [CrossRef]
39. Huang, H.; Shi, G.; He, H.; Duan, Y.; Luo, F. Dimensionality reduction of hyperspectral imagery based on spatial-spectral manifold learning. *IEEE Trans. Cybern.* **2019**, *50*, 2604–2616. [CrossRef] [PubMed]
40. Huang, H.; Duan, Y.; He, H.; Shi, G.; Luo, F. Spatial-spectral local discriminant projection for dimensionality reduction of hyperspectral image. *ISPRS J. Photogramm. Remote Sens.* **2019**, *156*, 77–93. [CrossRef]