Unsupervised Spectral–Spatial Feature Learning via Deep Residual Conv–Deconv Network for Hyperspectral Image Classification

Lichao Mou, *Student Member, IEEE*, Pedram Ghamisi, *Member, IEEE*, and Xiao Xiang Zhu, *Senior Member, IEEE*

***Abstract*— Supervised approaches classify input data using a set of representative samples for each class, known as *training samples*. The collection of such samples is expensive and time demanding. Hence, unsupervised feature learning, which has a quick access to arbitrary amounts of unlabeled data, is conceptually of high interest. In this paper, we propose a novel network architecture, fully Conv–Deconv network, for unsupervised spectral–spatial feature learning of hyperspectral images, which is able to be trained in an end-to-end manner. Specifically, our network is based on the so-called encoder–decoder paradigm, i.e., the input 3-D hyperspectral patch is first transformed into a typically lower dimensional space via a convolutional subnetwork (encoder), and then expanded to reproduce the initial data by a deconvolutional subnetwork (decoder). However, during the experiment, we found that such a network is not easy to be optimized. To address this problem, we refine the proposed network architecture by incorporating: 1) residual learning and 2) a new unpooling operation that can use memorized max-pooling indexes. Moreover, to understand the "black box," we make an in-depth study of the learned feature maps in the experimental analysis. A very interesting discovery is that some specific "neurons" in the first residual block of the proposed network own good description power for semantic visual patterns in the object level, which provide an opportunity to achieve "free" object detection. This paper, for the first time in the remote sensing community, proposes an end-to-end fully Conv–Deconv network for unsupervised spectral–spatial feature learning. Moreover, this paper also introduces an in-depth investigation of learned features. Experimental results on two widely used hyperspectral data, Indian Pines and Pavia University, demonstrate competitive performance obtained by the proposed methodology compared with other studied approaches.**

***Index Terms*— Convolutional network, deconvolutional network, hyperspectral image classification, residual learning, unsupervised spectral–spatial feature learning.**

Manuscript received January 7, 2017; revised May 4, 2017 and June 29, 2017; accepted August 3, 2017. Date of publication October 24, 2017; date of current version December 27, 2017. This work is jointly supported by the China Scholarship Council, Alexander von Humboldt Foundation, the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme (grant agreement No [ERC-2016-StG-714087], Acronym: *So2Sat*), and Helmholtz Association under the framework of the Young Investigators Group "SiPEO" (VH-NG-1018, www.sipeo.bgu.tum.de). *(Corresponding author: Xiao Xiang Zhu.)*

The authors are with the Remote Sensing Technology Institute, German Aerospace Center, 82234 Wessling, Germany, and also with Signal Processing in Earth Observation, Technical University of Munich, 80333 Munich, Germany (e-mail: lichao.mou@dlr.de; p.ghamisi@gmail.com; xiao.zhu@dlr.de).

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.2017.2748160

I. Introduction

ALONG with the development of different earth observation missions, hyperspectral imagery has been accessible at a reasonable cost over the last decade. Since hyperspectral images are characterized in hundreds of continuous observation bands, throughout the electromagnetic spectrum with high spectral resolution, such data have attracted considerable attention in the remote sensing community [1]. On the other hand, the analysis of hyperspectral images is of high importance in many practical applications, such as urban development [2]–[5], monitoring of land changes [6]–[9], and resource management [10], [11]. To benefit from these types of data, supervised hyperspectral image classification is among the most active research areas in hyperspectral analysis.

There is a vast literature on supervised classification models such as decision trees [12], random forests [13], [14], and support vector machines (SVMs) [15], [16]. A random forest [14] is an ensemble learning approach that operates by constructing several decision trees in the training course and outputting the classes of the input hyperspectral pixels via integration of predictions of the individual trees. In contrast, as a significant branch of the supervised machine learning algorithm, SVMs have achieved a great success in various applications due to the fact that they can handle high-dimensional data with a limited number of training samples. SVM works by mapping data to a kernel-included high-dimensional feature space seeking an optimal decision hyperplane that can best separate data samples, when data points are not linearly separable. SVM, therefore, has been considered to be an effective and stable algorithm for hyperspectral image classification task. In addition, some extensions of the SVM model [17], [18] have been proposed for hyperspectral data analysis to improve discrimination capability of the classifier. However, random forests and SVMs are attributed as "shallow" models, which means that their ability to deal with nonlinear data, e.g., hyperspectral data demonstrate dense nonlinearity, is limited compared with the "deep" ones.

With the investigation of hyperspectral image classification, a major finding is that various atmospheric scattering conditions, complicated light scattering mechanisms, interclass similarity, and intraclass variability result in hyperspectral imaging procedure being inherently nonlinear [19]. It is believed that, compared with the "shallow" models,

deep learning architectures are able to extract high-level, hierarchical, and abstract features, which are generally more robust to the nonlinear input data. So far, some studies in the community have focused on making use of deep learning models for hyperspectral image classification. For instance, Chen *et al.* [20] employed a stacked auto-encoder to extract hierarchical features from the spectral domain of hyperspectral images for the purpose of classification. In [21], a restricted Boltzmann machine (RBM) and its extension, deep belief network (DBN), were introduced for the classification of hyperspectral data by learning spectral-based features. Tao *et al.* [22] presented a multiscale sparse stacked auto-encoder to learn an effective feature representation from unlabeled data, and then the learned features were fed into a linear SVM for hyperspectral data classification. Very recently, Mou *et al.* [23] proposed a novel recurrent neural network with a new activation function and a modified gated recurrent unit for hyperspectral image classification, which can effectively analyze hyperspectral pixels as sequential data and then determine information categories via network reasoning.

Most of the aforementioned networks, e.g., auto-encoder, RBM, and DBN, are both early and fairly simple 1-D deep learning architectures totally equipped with fully connected layers. Consequently, there are a lot of trainable parameters that need to be estimated, which is an undesirable case given that available labeled training samples for remote sensing image classification are often limited [24]. Moreover, it should be noted that the processing mechanism of the 1-D networks and the vector-based feature alignment can lead to the loss of structure information for hyperspectral imagery, as it has an intrinsic 2-D data structure in the spatial domain.

Convolutional neural network (CNN), an important branch of the deep learning family, has been attracting attention, due to the fact that they are capable of automatically discovering relevant contextual 2-D spatial features in image categorization tasks. In addition, a CNN makes use of local connections to deal with spatial dependencies via sharing weights, and thus can significantly reduce the number of parameters of the network in comparison with the conventional 1-D fully connected neural networks. CNNs have already outperformed other methodologies in various domains of machine learning and computer vision such as large-scale natural image recognition [25]–[28], object detection [29], [30], and scene interpretation [31]–[35]. Very recently, a few supervised CNN-based models have been proposed for spectral–spatial classification of hyperspectral remote sensing images. Chen *et al.* [36] introduced a supervised ℓ_2 regularized 3-D CNN-based feature extraction model to extract efficient spectral–spatial features for the purpose of classification. Ghamisi *et al.* [19] proposed a self-improving CNN (SICNN) model, which combined a CNN with a fractional order Darwinian particle swarm optimization (FODPSO) algorithm to iteratively select the most informative bands suitable for training the designed CNN. Makantasis *et al.* [37] exploited a CNN to encode spectral and spatial information of input hyperspectral data followed by a multilayer perceptron to conduct the hyperspectral image classification task. Zhao and Du [38] proposed a spectral–spatial feature-based classification framework, which jointly makes use of a local discriminant embedding-based dimension reduction algorithm and a CNN for the purpose of land cover classification. Aptoula *et al.* [39] fed attribute profile features instead of original hyperspectral data into a CNN, which led to a performance improvement.

Those CNNs trained in a supervised manner via backpropagation, which improved the state-of-the-art performance on the hyperspectral image classification task. Despite the big success of the supervised CNNs, they have at least one potential drawback detailed as follows: there is a need for a good supply of labeled training samples to be used for supervised training. However, these are difficult to collect, and there are diminishing returns of making the labeled data set larger and larger. In other words, the supervised CNNs generally suffer from either small number of training samples or imbalanced data sets.

Hence, unsupervised spectral–spatial feature learning, which has a quick access to arbitrary amounts of unlabeled data, is conceptually of high interest. In general, the main purpose of unsupervised feature learning is to extract useful features from unlabeled data, to detect and remove input redundancies, and to preserve only essential aspects of the data in robust and discriminative representations. In a pioneer work moving from the supervised CNN to unsupervised CNN, Romero *et al.* [40] proposed an unsupervised convolutional network for learning spectral–spatial features using sparse learning to estimate the weights of the network. However, this model was trained in a greedy layer-wise fashion, i.e., it is not an end-to-end network. In this paper, we aim to propose an end-to-end network, fully Conv–Deconv network, for unsupervised spectral–spatial feature learning of hyperspectral imagery. Basically, our network architecture is based on the so-called encoder–decoder paradigm. Specifically, the input is first transformed into a typically lower dimensional space via a convolutional subnetwork (encoder), and then expanded to reproduce the initial data by a deconvolutional subnetwork (decoder). Moreover, the trained unsupervised Conv–Deconv network can be adapted to the classification of hyperspectral data by cutting off the deconvolutional subnetwork, replacing the loss function, and fine-tuning it to the new task, i.e., adjusting the weights using backpropagation. With this approach, typically much smaller training sets are sufficient. In detail, our work contributes to the literature in three major aspects.

1) We propose an end-to-end deep Conv–Deconv neural network, which is composed of a convolutional subnetwork and a deconvolutional subnetwork with a specially designed unpooling layer. Learning such a 2-D encoder–decoder-based network for unsupervised spectral–spatial feature learning of hyperspectral data has not been investigated yet to the best of our knowledge.
2) Since our network is fairly deep, it might easily converge to an inappropriate solution if small learning rates are used. On the other hand, simply boosting convergence with high learning rates leads to exploding the gradient problem. In this paper, we resolve this issue by introducing residual learning in our Conv–Deconv network. To the best of our knowledge, this is the first

use of residual learning to train networks for remote sensing data analysis.

3) Our unsupervised network is able to preserve the neighborhood relations and spatial locality of 3-D hyperspectral cubes in its latent high-level feature representations, while the conventional 1-D fully connected unsupervised network architectures such as auto-encoder, RBM, and DBN do not scale well to realistic-sized high-dimensional hyperspectral data in terms of computational complexity.
4) To understand the "black box" of the proposed network, we make an in-depth investigation. We found that some specific "neurons" in the first residual block of the network are capable of precisely capturing semantic visual patterns in object level, which makes it possible to achieve a high-quality unsupervised object detection capability for hyperspectral images.

The rest of this paper is organized as follows. An introduction to the traditional unsupervised network architectures is briefly given in Section II. The details of the proposed fully Conv–Deconv network with residual learning for unsupervised spectral–spatial feature extraction of hyperspectral data are described in Section III. The network setup, network analysis, experimental results, and a comparison with state-of-the-art approaches are provided in Section IV. Finally, Section V concludes this paper.

II. Preliminaries

Several types of traditional 1-D unsupervised network architectures have been leveraged for feature learning of hyperspectral data. In this section, we recall the basic principles of such models.

A. Auto-Encoder

An auto-encoder [41] takes an input $\boldsymbol{x} \in \mathbb{R}^D$ and first maps it to a latent representation $\boldsymbol{h} \in \mathbb{R}^M$ via a nonlinear mapping

$$\boldsymbol{h} = f(\boldsymbol{\Theta}\boldsymbol{x} + \boldsymbol{\beta}) \tag{1}$$

where $\boldsymbol{\Theta}$ is a weight matrix to be estimated during the training course, $\boldsymbol{\beta}$ is a bias vector, and f stands for a nonlinear function such as the logistic sigmoid function and hyperbolic tangent function. The encoded feature representation $\boldsymbol{h}$ is then used to reconstruct the input $\boldsymbol{x}$ by a reverse mapping

$$\boldsymbol{y} = f(\boldsymbol{\Theta}'\boldsymbol{h} + \boldsymbol{\beta}') \tag{2}$$

where $\boldsymbol{\Theta}'$ is usually constrained to be the form of $\boldsymbol{\Theta}' = \boldsymbol{\Theta}^T$, using the same weight for encoding the input and decoding the latent representation. The reconstruction error is defined as the Euclidian distance between $\boldsymbol{x}$ and $\boldsymbol{y}$ that is constrained to approximate the input data $\boldsymbol{x}$, i.e., making $\|\boldsymbol{x} - \boldsymbol{y}\|_2^2 \to 0$. The parameters of the auto-encoder are generally optimized by stochastic gradient descent (SGD) [42]. Fig. 1 illustrates the structure of the auto-encoder.

B. Sparse Auto-Encoder

The conventional auto-encoder relies on the dimension of the latent representation $\boldsymbol{h}$ being smaller than that of input $\boldsymbol{x}$, i.e., $M < D$, which means it tends to learn a low-dimensional compressed representation. However, when $M > D$, one can still discover an interesting structure, by enforcing a sparsity constraint on the hidden units. Formally, given a set of unlabeled data $\boldsymbol{X} = \{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^N\}$, training a sparse auto-encoder is to find the optimal parameters by minimizing the following loss function:

$$\mathbb{E} = \frac{1}{N}\sum_{i=1}^{N}\left(J(\boldsymbol{x}^i, \boldsymbol{y}^i; \boldsymbol{\Theta}, \boldsymbol{\beta}) + \lambda \sum_{j=1}^{M} \mathrm{KL}(\rho \| \hat{\rho}_j)\right) \tag{3}$$

where $J(\boldsymbol{x}^i, \boldsymbol{y}^i; \boldsymbol{\Theta}, \boldsymbol{\beta})$ is an average sum-of-squares error term, which represents the reconstruction error between the input $\boldsymbol{x}^i$ and its reconstruction $\boldsymbol{y}^i$. $\mathrm{KL}(\rho \| \hat{\rho}_j)$ is the Kullback-Leibler (KL) divergence between a Bernoulli random variable with mean ρ and a Bernoulli random variable with mean $\hat{\rho}_j$. KL divergence is a standard function for measuring how similar two distributions are, and it can be described as follows:

$$\mathrm{KL}(\rho \| \hat{\rho}_j) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1-\rho)\log \frac{1-\rho}{1-\hat{\rho}_j}. \tag{4}$$

In the sparse auto-encoder model, KL divergence is called sparsity penalty term that provides the sparsity constraint, and λ controls the weight of the sparsity penalty term. Similar to the auto-encoder, the optimization of a sparse auto-encoder can be achieved via the backpropagation and SGD [42].

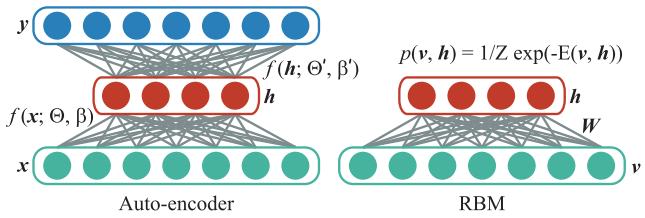


Fig. 1. Two classical unsupervised network architectures. (Left) Auto-encoder. (Right) RBM.

C. RBM and DBN

Unlike the deterministic network architectures such as auto-encoder or sparse auto-encoder, an RBM is a stochastic undirected graphical model consisted of a visible layer and a hidden layer, and it has symmetric connections between these two layers, and no connecting exists within the hidden layer or the input layer. The energy function of an RBM can be defined as follows:

$$E(\boldsymbol{x}, \boldsymbol{h}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{x} - (\boldsymbol{h}^T\boldsymbol{W}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} + \boldsymbol{b}^T\boldsymbol{h}) \tag{5}$$

where $\boldsymbol{W}$, $\boldsymbol{c}$, and $\boldsymbol{b}$ are the weights of the RBM model. The joint probability distribution of the RBM is defined as

$$p(\boldsymbol{x}, \boldsymbol{h}) = \frac{1}{Z}\exp(-\mathbb{E}(\boldsymbol{x}, \boldsymbol{h})) \tag{6}$$

where Z is a normalization constant. The form of the RBM makes the conditional probability distribution computationally feasible, when $\boldsymbol{x}$ or $\boldsymbol{h}$ is fixed. The structure of the RBM is depicted in Fig. 1.

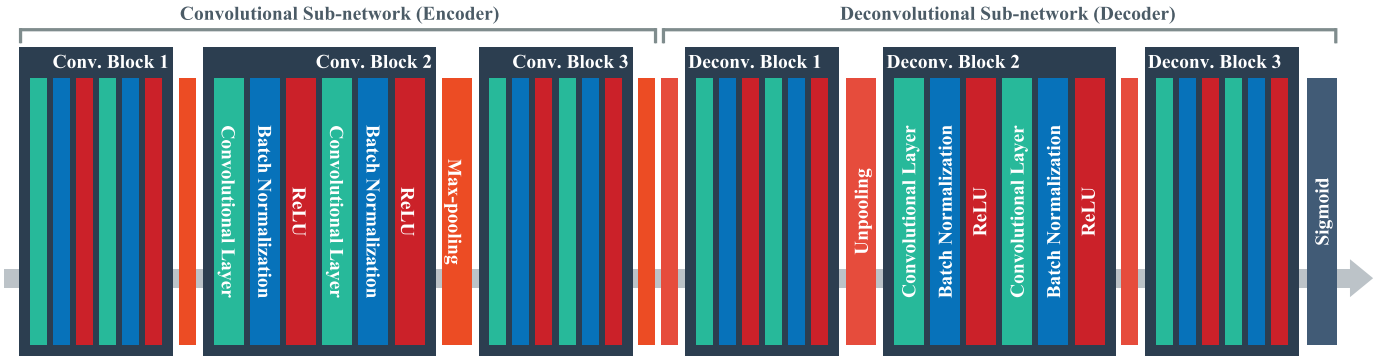


Fig. 2. We propose a network architecture that learns to extract spectral–spatial features by reconstructing the initial input 3-D hyperspectral patches, being trained end to end. There are no fully connected layers, and hence it is a fully Conv–Deconv network. The proposed network architecture is composed of two parts, i.e., convolutional subnetwork and deconvolutional subnetwork. The former corresponds to an encoder that transforms the input 3-D hyperspectral patches to abstract feature representations, whereas the latter plays the role of decoder that reproduces the initial input data from the encoded features. Each layer in the convolutional subnetwork has a corresponding decoder layer in the deconvolutional subnetwork.

The feature representation ability of a single RBM is limited. However, its real power emerges when a couple of RBMs are stacked, forming a DBN [43]. Hinton *et al.* [43] proposed a greedy approach that trains RBM in each layer to efficiently train a DBN.

III. Methodology

CNNs have shown to be very successful on a range of visual recognition tasks [25]–[27], [29]–[33]. Such tasks, in common, can be posed as discriminative supervised learning problems, and hence, can be resolved by CNNs, which are well known to be effective at learning input–output relations given an adequate number of labeled data sets. Normally, a task solved by making use of CNNs involves learning mappings from concrete raw images to some sort of condensed abstract outputs, such as category. Here, we are interested in training an end-to-end neural network to extract features in an unsupervised fashion, which means we need to leverage a network to solve a concrete-to-concrete problem instead of the traditional concrete-to-abstract one. This brings up a question in mind: what is a good network architecture for our purpose?

A. Initial Conv–Deconv Network Architecture

1) Analysis and Modeling: Denote by $(\boldsymbol{x}, \boldsymbol{h}, \boldsymbol{y})$ random variables representing a 3-D hyperspectral patch, its encoded feature representation, and the reconstructed output. The joint probability distribution $p(\boldsymbol{x}, \boldsymbol{y})$ can be described as follows:

$$p(\boldsymbol{x}, \boldsymbol{y}) = p(\boldsymbol{x})p(\boldsymbol{y}|\boldsymbol{x}) \tag{7}$$

where $p(\boldsymbol{x})$ is the distribution of 3-D hyperspectral patches and $p(\boldsymbol{y}|\boldsymbol{x})$ is the distribution of reconstructed outputs given the hyperspectral patches. Thus, the conditional probability distribution $p(\boldsymbol{y}|\boldsymbol{x})$ can be written as

$$p(\boldsymbol{y}|\boldsymbol{x}) = p(\boldsymbol{y}, \boldsymbol{h}|\boldsymbol{x}) = p(\boldsymbol{y}|\boldsymbol{h})p(\boldsymbol{h}|\boldsymbol{x}) \tag{8}$$

where $p(\boldsymbol{h}|\boldsymbol{x})$ indicates the distribution of the encoded feature representations given the input hyperspectral patches. As a special case, $\boldsymbol{y}$ may be a deterministic function of $\boldsymbol{x}$. Ideally, we would like to find $p(\boldsymbol{h}|\boldsymbol{x})$ and $p(\boldsymbol{y}|\boldsymbol{h})$, but direct application of Bayesian theory is not feasible. We, therefore, in this paper resort to an estimate function $f(\boldsymbol{x})$ that minimizes the following mean squared error objective:

$$\mathbb{E}_{\boldsymbol{x}} \|\boldsymbol{x} - f(\boldsymbol{x})\|_2^2. \tag{9}$$

The minimizer of this loss is the conditional expectation

$$\hat{f}(\boldsymbol{x}_0) = \mathbb{E}_{\boldsymbol{y}}[\boldsymbol{y}|\boldsymbol{h}] + \mathbb{E}_{\boldsymbol{h}}[\boldsymbol{h}|\boldsymbol{x} = \boldsymbol{x}_0] \tag{10}$$

that is the expected reconstructed output given a hyperspectral patch.

Given a set of unlabeled 3-D hyperspectral patches $\{\boldsymbol{x}_i\}$, we learn the weights $\boldsymbol{\Theta}$ of a network $f(\boldsymbol{x}; \boldsymbol{\Theta})$ to minimize a Monte Carlo estimate of the loss (9)

$$\hat{\boldsymbol{\Theta}} = \arg\min_{\boldsymbol{\Theta}} \sum_i \|\boldsymbol{x}_i - f(\boldsymbol{x}_i; \boldsymbol{\Theta})\|_2^2. \tag{11}$$

This means that we train the network to reproduce the input results in learning high-level abstract features in an unsupervised manner.

In this paper, we propose a fully Conv–Deconv network (see Fig. 2) in which the desired output is the input data itself. The proposed network architecture is composed of two parts, i.e., the convolutional subnetwork and deconvolutional subnetwork. The former corresponds to an encoder that transforms the input 3-D hyperspectral patch $\boldsymbol{x}_i$ to abstract feature representation $\boldsymbol{h}_i$, whereas the latter plays the role of a decoder that reproduces the initial input data from the encoded feature. Each layer in the convolutional subnetwork has a corresponding decoder layer in the deconvolutional subnetwork.

2) Convolutional Subnetwork: The design of the architecture of the convolutional subnetwork is mainly inspired by the philosophy of the VGG Nets [26]. The input hyperspectral patch is fed into a stack of convolutional layers, where we leverage convolutional filters with a very small receptive field of 3×3, rather than making use of larger ones, such as 5×5 or 7×7. The reason is that 3×3 convolutional filters are the smallest kernels to seize patterns in different directions, such as center, up/down, and left/right, but still have an advantage: the usage of small convolutional filters will increase the nonlinearities inside the network and thus make the network more discriminative.

In addition, the convolutional stride in the convolutional subnetwork is fixed to 1 pixel; the spatial padding of convolutional layer input is such that the spatial resolution of feature maps is preserved after convolution, in other words, the padding is 1 pixel for the used 3×3 convolutional layers. Spatial pooling is achieved by carrying out several max-pooling layers, which follow some of the convolutional layers. In particular, max pooling is performed over 3×3 pixel windows with stride 3.

In a nutshell, the convolutional layers in the convolutional subnetwork consist of 3×3 filters and follow the following two rules: 1) the convolutional layers in each convolutional block are with the same feature map size and have the same number of filters and 2) the number of channels of the feature maps increases in the deeper convolutional blocks, roughly doubling after each max-pooling layer, which is meant to preserve the time complexity per layer as far as possible. All layers in the convolutional subnetwork are equipped with a rectified linear unit (ReLU) [25] as activation function. ReLU is one of several keys to the recent success of deep neural networks and can be defined as $f(x) = \max(0, x)$. Compared with the conventional activation functions, such as sigmoid and hyperbolic tangent function, the usage of ReLU can expedite convergence of the training course and result in better solutions.

3) Deconvolutional Subnetwork: The convolutional subnetwork is in charge of extracting high-level abstract spectral–spatial feature representation of the input 3-D hyperspectral patch, by interleaving convolutional layers and max-pooling layers, i.e., spatially shrinking the feature maps layer by layer. Pooling is necessary to allow agglomerating information over large areas of feature maps and, more fundamentally, to make the network computationally feasible. However, pooling leads to reduced resolution of the feature maps; hence, in order to reconstruct the initial input data, we need to find a way to refine this coarse pooled representation.

Our approach to this refinement is to construct a deconvolutional subnetwork. The main ingredient is deconvolutional operation, which performs reverse operation of the convolutional subnetwork and reconstructs the original input data from the abstract feature representation. The deconvolutional operation consists of unpooling and convolution. In order to map the encoded feature to a high-dimensional hyperspectral cube, we need unpooling to unpool the feature maps, i.e., to increase their spatial span, as opposed to the pooling (spatially shrinking the feature maps) implemented by the convolutional subnetwork. More specifically, the unpooling [44], [45] is performed by simply replacing each entry of a feature map by an $s \times s$ block with the entry value in the top-left corner and zeros elsewhere (see Fig. 3). With this operation, the height and the width of the feature maps are increased s times. In this network, we made use of $s = 3$, as the size of the receptive field in the max-pooling layers of the convolutional subnetwork is 3×3. When a convolutional block is preceded by an unpooling layer, we can thus think of the combination of unpooling and convolutional block as the inverse operation of "convolutional block + pooling" performed in the convolutional subnetwork.

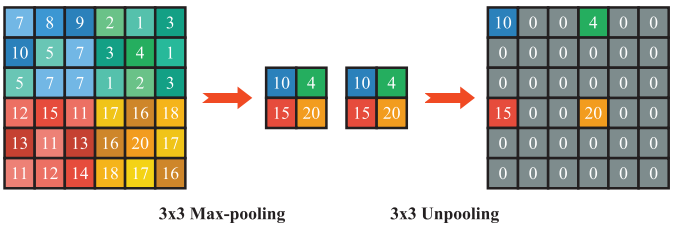


Fig. 3. Illustration of (Left) max pooling and (Right) unpooling as used in the fully Conv–Deconv network described in Section III-A.

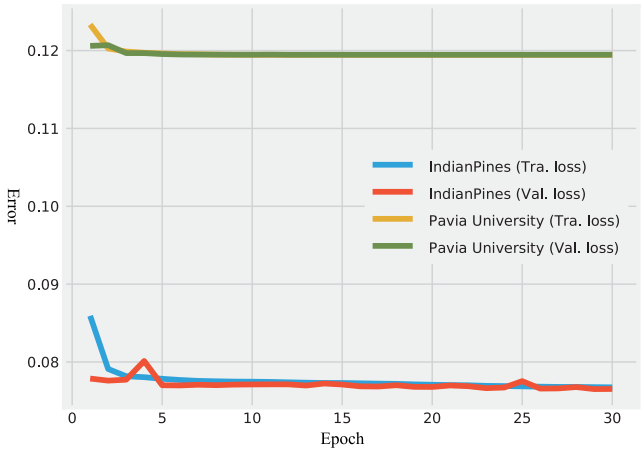


Fig. 4. Learning curves for the initial fully Conv–Deconv network on the Indian Pines data set and the Pavia University data set. Although the network starts greatly reducing errors on both the training and validation samples during the first few epochs, it rapidly converges to a fairly high value, which means the learning of the network is significantly slowed down and eventually gets stuck into a local minimum. This indicates that such a network architecture is not easy to optimize.

The configuration of convolutional blocks in the deconvolutional subnetwork is the same with the convolutional subnetwork, namely, 3×3 receptive field, 1 pixel padding, and ReLU as activation function.

B. Refined Network Architecture

1) Difficulty of Training Conv–Deconv Network: In Section III-A, we have systematically built a reasonable network architecture for our task, but a problem will arise when we attempt to train the network. As can be seen in Fig. 4, although the network starts greatly reducing errors on both the training and validation samples during the first few epochs, it rapidly converges to a fairly high value, which means the learning of the network is significantly slowed down and eventually gets stuck into a local minimum. This indicates that such network architecture is not easy to optimize. We think the obstacles to train the proposed fully Conv–Deconv network are as follows.

1) In the Conv–Deconv network, the exact copy of the input high-dimensional 3-D hyperspectral patch has to go through all layers until it reaches the output layer. With many weight layers, this becomes an end-to-end relation requiring very long-term memory. For this reason, the notorious vanishing gradient problem [46], [47] can

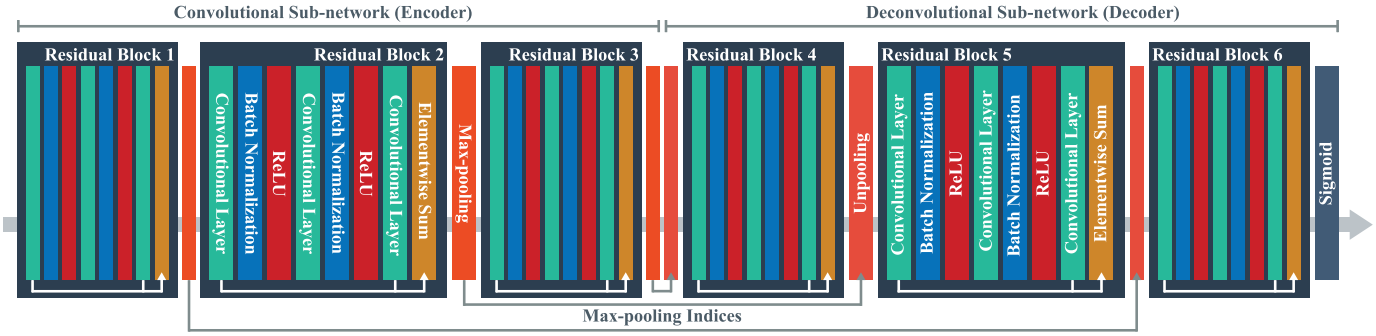


Fig. 5. We refine the proposed fully Conv–Deconv network architecture by incorporating residual learning and a more appropriate unpooling operation, which can use memorized max-pooling indices from the corresponding encoded feature maps and enables reconstruction to be more accurate.

be critical, which handicaps the learning process of the network.

2) The unpooling operation [44], [45] in the deconvolutional subnetwork increases the spatial resolution of feature maps by simply adding zeros, which ignores the location of the maximum value in the receptive field of pooling layer, leading to loss of edge information during the decoding procedure. Without this detailed information, it is difficult for the optimizer to lead the network to better solutions.

To address the aforementioned problems, in this section, we refine the proposed fully Conv–Deconv network architecture by incorporating residual learning and a new unpooling operation that can use memorized max-pooling indices from the corresponding encoded feature maps and enables reconstruction to be more accurate. The new network architecture is shown in Fig. 5.

2) Conv–Deconv Network With Residual Learning: Residual learning has recently shown appealing performance in the concrete-to-abstract deep network architectures on many challenging visual tasks, such as image classification [27], [48] and object detection [27]. One main merit offered using the residual learning is that it helps in handling the vanishing gradient problem and degradation problem [27]. In this paper, we are interested in introducing the residual learning to the proposed concrete-to-concrete Conv–Deconv network in order to resolve the network training problem.

The proposed Conv–Deconv network with residual learning is a modularized network architecture that stacks residual blocks. Similar to the convolutional blocks, a residual block consists of several convolutional layers that are with the same feature map size and have the same number of filters. However, it performs the following calculation:

$$\boldsymbol{\varphi}_l = g(\boldsymbol{\phi}_l) + \mathcal{F}(\boldsymbol{\phi}_l; \boldsymbol{\Theta}_l) \tag{12}$$

$$\boldsymbol{\phi}_{l+1} = f(\boldsymbol{\varphi}_l). \tag{13}$$

Here, $\boldsymbol{\phi}_l$ indicates the feature maps that are fed into the lth residual block and satisfies $\boldsymbol{\phi}_0 = \boldsymbol{x}$ where $\boldsymbol{x}$ is the input 3-D hyperspectral patch. $\boldsymbol{\Theta}_l = \{\boldsymbol{\Theta}_{l,k} | 1 \leq k \leq K\}$ represents a collection of weights associated with the lth residual block, and K denotes that there are K convolutional layers in a residual block. Moreover, $\mathcal{F}$ is the residual function and is generally achieved by few stacked convolutional layers, e.g., a convolutional block described in Section III-A. The function f indicates the activation function such as a linear activation function or ReLU, and f works after element-wise addition. The function g is fixed to an identity mapping: $g(\boldsymbol{\phi}_l) = \boldsymbol{\phi}_l$.

If f adopts a linear activation function and also acts as an identity mapping, i.e., $\boldsymbol{\phi}_{l+1} = \boldsymbol{\varphi}_l$, we can obtain the output of the lth residual block by putting (12) into (13)

$$\boldsymbol{\phi}_{l+1} = \boldsymbol{\phi}_l + \mathcal{F}(\boldsymbol{\phi}_l; \boldsymbol{\Theta}_l). \tag{14}$$

In contrast, a convolutional block only performs the following computation:

$$\boldsymbol{\phi}_{l+1} = \mathcal{H}(\boldsymbol{\phi}_l; \boldsymbol{\Theta}_l). \tag{15}$$

Recursively, like

$$\begin{aligned} \boldsymbol{\phi}_{l+2} &= \boldsymbol{\phi}_{l+1} + \mathcal{F}(\boldsymbol{\phi}_{l+1}; \boldsymbol{\Theta}_{l+1}) \\ &= \boldsymbol{\phi}_l + \mathcal{F}(\boldsymbol{\phi}_l; \boldsymbol{\Theta}_l) + \mathcal{F}(\boldsymbol{\phi}_{l+1}; \boldsymbol{\Theta}_{l+1}) \end{aligned} \tag{16}$$

we will get the following recurrence formula:

$$\boldsymbol{\phi}_L = \boldsymbol{\phi}_l + \sum_{i=l}^{L-1} \mathcal{F}(\boldsymbol{\phi}_i; \boldsymbol{\Theta}_i) \tag{17}$$

for any shallower block l and any deeper block L.

As exhibited in (17), the network with residual learning has some interesting and nice properties.

1) The feature maps $\boldsymbol{\phi}_L$ of any deeper residual block L can be considered to be adding the feature maps $\boldsymbol{\phi}_l$ of any shallower block l and a residual function in a form of $\sum_{i=1}^{L-1} \mathcal{F}$, representing that the network is in a residual fashion and is capable of learning some new features between any blocks l and L.
2) With both the g and f being identity mappings, i.e., $g(\boldsymbol{\phi}_l) = \boldsymbol{\phi}_l$ and $f(\boldsymbol{\varphi}_l) = \boldsymbol{\varphi}_l$, a network with residual learning creates a direct path for propagating information through the entire network, which can effectively avoid the vanishing gradient problem.

These two respects are in contrast to the Conv–Deconv network equipped with common convolutional blocks (see Section III-A) in which the feature maps $\boldsymbol{\phi}_L$ are a set of matrix products, namely, $\prod_{i=0}^{L-1} \boldsymbol{\Theta}_i \boldsymbol{\phi}_0$.

The content discussed above illustrates the forward propagation procedure of the Conv–Deconv network with residual

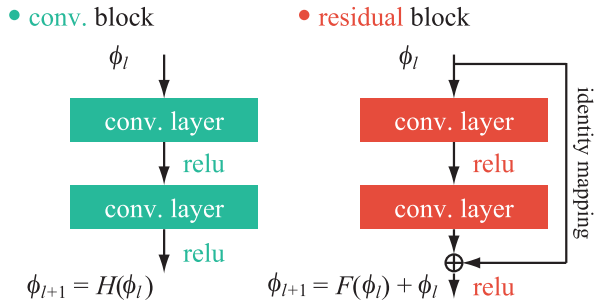


Fig. 6. Comparison between the convolutional block and the residual block. Here, $\boldsymbol{\phi}_l$ denotes the input and $\boldsymbol{\phi}_{l+1}$ is any desired output. The convolutional block hopes that two convolutional layers are able to fit $\boldsymbol{\phi}_{l+1}$ by directly learning a mapping $\mathcal{H}$. In contrast, the two convolutional layers are expected to learn a residual function $\mathcal{F}$ to let $\boldsymbol{\phi}_{l+1} = \mathcal{F}(\boldsymbol{\phi}_l) + \boldsymbol{\phi}_l$ in the residual block.

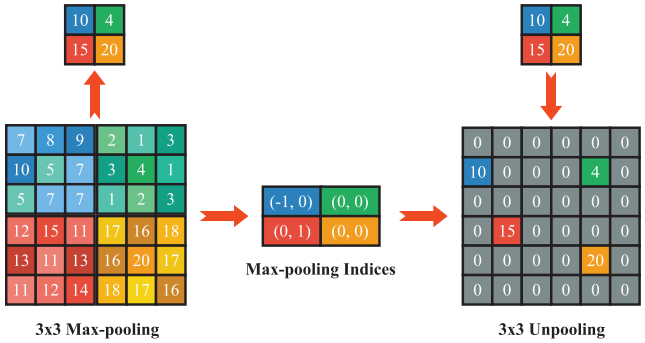


Fig. 7. Illustration of the unpooling operation in the refined Conv–Deconv network (see Section III-B), using max-pooling indices that are capable of recording the location of the maximum value in each local pooling region during pooling in the convolutional subnetwork.

learning. However, how the residual learning can help us to effectively train the proposed deep network? To answer this question, we need to dive into the backward propagation process. Denoted by $\mathbb{E}$ indicating the loss function, according to the chain rule of backpropagation, we can obtain

$$\frac{\partial \mathbb{E}}{\partial \boldsymbol{\phi}_l} = \frac{\partial \mathbb{E}}{\partial \boldsymbol{\phi}_L}\frac{\partial \boldsymbol{\phi}_L}{\partial \boldsymbol{\phi}_l} = \frac{\partial \mathbb{E}}{\partial \boldsymbol{\phi}_L}\left(1 + \frac{\partial}{\partial \boldsymbol{\phi}_l}\sum_{i=l}^{L-1}\mathcal{F}(\boldsymbol{\phi}_i; \boldsymbol{\Theta}_i)\right). \quad (18)$$

Equation (18) implies that the gradient $(\partial\mathbb{E}/\partial\boldsymbol{\phi}_l)$ can be decomposed into two additive terms: a term of $(\partial\mathbb{E}/\partial\boldsymbol{\phi}_L)$ that directly propagates information without concerning any weight layers and another term of $(\partial\mathbb{E}/\partial\boldsymbol{\phi}_L)((\partial/\partial\boldsymbol{\phi}_l)\sum_{i=l}^{L-1}\mathcal{F})$ that propagates through the weight layers. The former term ensures that the information can be propagated back to any shallower residual block l directly. In addition, since $(\partial/\partial\boldsymbol{\phi}_l)\sum_{i=l}^{L-1}\mathcal{F}$ basically cannot always be -1 for all training data in a batch, it is almost impossible that (18) is canceled out for a mini-batch. This implies that the gradient information of a layer in the network does not vanish even while the trainable weights are arbitrarily small, which is the key to make the deep network feasible for the purpose of training and to answer the question mentioned above. Given the activation function of the last layer is sigmoid, on the contrary, the initial Conv–Deconv network easily suffers from the vanishing gradient problem, which leads the learning procedure is slowed down or even stopped. Fig. 6 shows a comparison between the convolutional block [Fig. 6 (left)] and the residual block [Fig. 6 (right)].

3) More Accurate Unpooling: To acquire more appropriate unpooled feature maps and more precise reconstruction output, the max-pooling indices computed in the max-pooling layers of the corresponding encoder can be used to perform nonlinear upsampling of the feature maps. And, reusing the max-pooling indices in the deconvolutional subnetwork has several practical merits, including that it is able to improve boundary delineation and eliminates the need for learning to upsample. The unpooled feature maps produced by this form of unpooling are sparse. Then, the unpooled feature maps are convolved with trainable filters to generate dense feature maps.

Goroshin *et al.* [49] recently presented a soft version of max and arg max operations that can take not only the maximum value in the receptive field of a max-pooling layer but also the corresponding index of that value. In particular, these two operations can be computed as follows:

$$\mu = \sum_{\mathcal{V}} z(i,j)\frac{\exp(\alpha z(i,j))}{\sum_{\mathcal{V}}\exp(\alpha z(i,j))} \approx \max_{\mathcal{V}} z(i,j) \quad (19)$$

$$\nu = \sum_{\mathcal{V}} [i,j]^T \frac{\exp(\alpha z(i,j))}{\sum_{\mathcal{V}}\exp(\alpha z(i,j))} \approx \arg\max_{\mathcal{V}} z(i,j) \quad (20)$$

where (i, j) stands for the spatial location index in the receptive field of a max-pooling layer and takes normalized values from -1 to 1, and $z(i, j)$ presents the value of the given location on a feature map. $\mathcal{V}$ is the receptive field. Note that α is a hyperparameter that controls soft pooling such that the lager the α, the closer the soft pooling approaches max pooling. With the max and arg max operations, the max-poling indices can be obtained in every pooling layer.

Then we make use of interpolation in the unpooling layers of the deconvolutional subnetwork by handling the values conveyed by the max-pooling indices (see Fig. 7). The use of max-pooling indices enables location information to be more accurately represented and thus enables the feature maps to capture fine details about the input 3-D hyperspectral patch.

C. Usage of Learned Features for Classification by Fine-Tuning the Network

Once the Conv–Deconv network is trained, the convolutional subnetwork, i.e., the encoder, can be regarded as an effective feature extractor. The key idea, here, is that the internal layers of the convolutional subnetwork can act as a generic extractor of spectral–spatial representation, which, first, can be trained by learning an identity mapping in the encoder–decoder architecture and then reused on other target tasks like classification. With this fine-tuning, we do not have to use a large number of labeled data to train a valid network for the purpose of supervised classification. In contrast, taking into consideration the fact that the total number of trainable parameters of a deep 2-D convolutional network is huge, a direct learning of so many parameters from the limited number of training samples is problematic. For fine-tuning, we cut off the deconvolutional subnetwork, introduce a new fully connected layer with softmax as a classifier, and fine-tune this

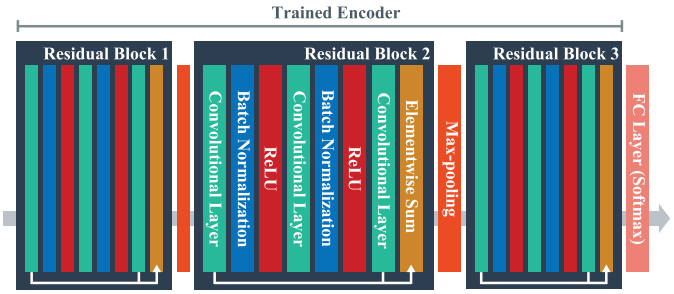


Fig. 8. Illustration of fine-tuning.

TABLE I
NUMBER OF TRAINING AND TEST SAMPLES USED IN THE INDIAN PINES DATA SET

Class No.	Class Name	Training	Test
1	Alfalfa	50	1384
2	Corn-notill	50	784
3	Corn-min	50	184
4	Corn	50	447
5	Grass-pasture	50	697
6	Grass-trees	50	439
7	Grass-pasture-mowed	50	918
8	Hay-windrowed	50	2418
9	Oats	50	564
10	Soybean-notill	50	162
11	Soybean-mintill	50	1244
12	Soybean-clean	50	330
13	Wheat	50	45
14	Woods	15	39
15	Buildings-grass-trees	15	11
16	Stone-steel-towers	15	5
	TOTAL	695	9671

new layer with limited labeled training samples, making the network significantly easier to be trained for the classification task. Fig. 8 illustrates this process.

IV. EXPERIMENTAL RESULTS AND DISCUSSION

A. Data Description

1) Indian Pines: This data set was acquired over the Indian Pines agricultural site in northwestern Indiana. It was collected with an airborne visible/infrared imaging spectrometer (AVIRIS) sensor in June 1992. The AVIRIS sensor comprises 220 spectral channels ranging from 400 to 2500 nm. In this data set, 20 bands affected by atmosphere absorption have been removed, and the remaining 200 spectral bands are investigated in this paper. The data set consists of 145 $\times$ 145 pixels, and the spatial resolution is 20 m/pixel. The available training samples of this data set cover 16 classes of interests. Table I provides information about different classes and their corresponding training and test samples.

2) Pavia University: The second data set was captured by reflective optics system imaging spectrometer (ROSIS) covering the Engineering School at the University of Pavia, and presents nine classes, mostly related to land covers. The image is of 610 $\times$ 340 pixels with a spatial resolution of 1.3 m/pixel and was collected under the HySens project managed by the German Aerospace Center. The hyperspectral imagery consists of 115 spectral channels ranging from 430 to 860 nm. In this paper, we made use of 103 spectral channels, after removing 12 noisy bands. Table II provides information about all nine classes of this data set with their corresponding training and test samples.

TABLE II
NUMBER OF TRAINING AND TEST SAMPLES USED IN THE PAVIA UNIVERSITY DATA SET

Class No.	Class Name	Training	Test
1	Asphalt	548	6631
2	Meadows	540	18649
3	Gravel	392	2099
4	Trees	524	3064
5	Metal sheets	265	1345
6	Bare Soil	532	5029
7	Bitumen	375	1330
8	Bricks	514	3682
9	Shadows	231	947
	TOTAL	3921	42776

B. General Information

To evaluate the performance of different approaches for hyperspectral image classification, the following evaluation criteria are used.

1) *Overall Accuracy (OA):* This measure represents the number of samples that are classified correctly, divided by the number of test samples.
2) *Average Accuracy (AA):* This index shows the average value of the classification accuracies of all categories.
3) *Kappa Coefficient:* This metric is a statistical measurement that provides information regarding the amount of agreement between the ground truth map and the final classification map. It is the percentage agreement corrected by the level of agreement, which could be expected due to the chance alone. In general, it is considered to be a more robust index than a simple percent agreement calculation, since k takes into account the agreement occurring by chance [1].

In addition, in order to evaluate the significance of the classification accuracies obtained by different approaches, a statistical test is conducted. Since the samples that were used for two different classification approaches are not independent, we evaluate the significance of two classification results with McNemar's test, which is given by [50]

$$z_{12} = \frac{f_{12} - f_{21}}{\sqrt{f_{12} + f_{21}}}$$

where f_{ij} is the number of correctly classified samples in classification i and incorrectly in classification j. McNemar's test is based on the standardized normal test statistic and therefore, the null hypothesis, which is "no significant difference," is rejected at the widely used $p = 0.05$ ($|z| > 1.96$) level of significance.

To validate the effectiveness of the proposed network architecture for the purpose of hyperspectral image classification, the novel classification method is compared with the most widely used supervised models, random forest [13], [14] and SVMs [15], [16]. In addition, in this paper, the experiments

making use of other supervised deep learning methods such as 1-D CNN and 2-D CNN are also carried out to verify the validity of the proposed network. The approaches included in the comparison are summarized as follows.

1) *RF-200:* Random forest with 200 trees.
2) *SVM-RBF:* SVMs with an RBF kernel are implemented using the libsvm package.[1] Furthermore, fivefold cross-validation is taken into account to tune the hyperplane parameters.
3) *1-D CNN:* The network architecture of the 1-D CNN is designed as in [51] and includes an input layer, convolutional layer, max-pooling layer, fully connected layer, and output layer. The number of the convolutional filters is 20 for all data sets. The length of each convolutional filter and the pooling size are 11 and 3, respectively. Moreover, 100 hidden units are contained in the fully connected layer.
4) *2-D CNN:* We follow the architecture of the 2-D CNN as used in [36]. It contains three convolutional layers that are equipped with 4×4, 5×5, and 4×4 convolutional filters, respectively. The convolutional layers—apart from the last one—are followed by the max-pooling layers. In addition, the numbers of the convolutional filters for the convolutional layers are 32, 64, and 128, respectively.
5) *SICNN:* An SICNN model solves the curse of dimensionality and the lack of available training samples by iteratively selecting the most informative bands suitable for the designed network via FODPSO [19].
6) *Initial Conv–Deconv Network:* The fully Conv–Deconv network with the plain convolutional blocks and the unpooling operation implemented in [44] and [45] (see Section III-A).
7) *Residual Conv–Deconv Network:* Our final network architecture makes use of the residual blocks and a more accurate unpooling operation. Section III-B shows the details.

Note that, to make the proposed approach fully comparable with other supervised classifiers in the literature, we used the standard sets of training and test samples for the data sets.

The fully Conv–Deconv network was trained using the Adam algorithm [52], and all the suggested default parameters were used for all the following experiments. The number of convolutional filters increases toward deeper layers of the convolutional subnetworks: 64 for the first residual block, 128 for the following block, and 256 for the last one. This rule is turned over for the deconvolutional subnetwork. All the convolutional layers are with ReLU as nonlinear activation function except the last layer that uses sigmoid activation. All weight matrices in the network and bias vectors are initialized with a uniform distribution, and the values of these weight matrices and bias vectors are initialized in the range $[-0.1, 0.1]$. The number of unlabeled data samples used for training the Conv–Deconv network on both Indian Pines and Pavia University is 10 000. These unlabeled samples are randomly selected from the whole images. Prior to training the Conv–Deconv network, we normalize the hyperspectral data in the range of 0–1. Then, all the weights can be updated during the training procedure. Once the training of Conv–Deconv network is complete, we can start to fine-tune the network for hyperspectral data classification. We made use of SGD with a fairly low learning rate of 0.0001 in order to fine-tune the network. For fine-tuning, in both hyperspectral data sets, we randomly chose 10% of the training samples as the validation set. That is, during fine-tuning, we used 90% of the training samples to learn the parameters and the remaining 10% of the training samples as validation to tune the super-parameters, such as the numbers of convolutional filters in the convolutional layers. All test samples are used to evaluate the final performance of the learned spectral–spatial feature representations and the fine-tuned network for classification.

The experiments are organized into three parts. The first part aims primarily at evaluating the learning procedures of the initial Conv–Deconv network and the residual Conv–Deconv network. Moreover, the learned feature maps are also shown and discussed in this part. In the second part, the effectiveness of the proposed network is compared with other state-of-the-art models such as random forest, SVM, 1-D CNN, and 2-D CNN. In the last part, we comment on the processing time.

C. Analysis of the Conv–Deconv Networks

1) Learning Curves: We first investigate the behavior of the initial Conv–Deconv network and the residual Conv–Deconv network during the training process, before we present the performance of the networks for the classification task. The qualities of the trained networks can be reflected by learning curves. As shown in Fig. 9, the initial Conv–Deconv network starts reducing error earlier on both the training samples and the validation samples but finally reduces the loss to a relatively high value, which means the learning of the network is apparently slowed down and the network converges to a local minimum in the end. In contrast, with residual learning, the residual Conv–Deconv network shows strong convergence ability. In particular, the residual Conv–Deconv network can obtain the training error value of 0.000276 on the Indian Pines data set after 30 epochs, while the initial Conv–Deconv network can achieve only 0.0767. For the Pavia University data set, the residual Conv–Deconv network can quickly converge to the error of 0.000238 after 30 iterations. In the same condition, the initial Conv–Deconv network can yield only 0.120. Furthermore, since we do not observe the overfitting problem in Fig. 9, the trained residual Conv–Deconv network can be thought as a good model for the follow-up fine-tuning stage.

2) Feature Visualization and Analysis: In order to understand the "black box" of the Conv–Deconv network, we show and analyze the learned feature maps. Specifically, we use the Pavia University data set to perform an in-depth study of the learned feature representation. Note that we do not have any fully connected layer in the residual Conv–Deconv network, which allows the trained network to take hyperspectral images of arbitrary size as input. Fig. 10 shows feature visualizations from the first residual block of the residual Conv–Deconv

[1] https://www.csie.ntu.edu.tw/ cjlin/libsvm/

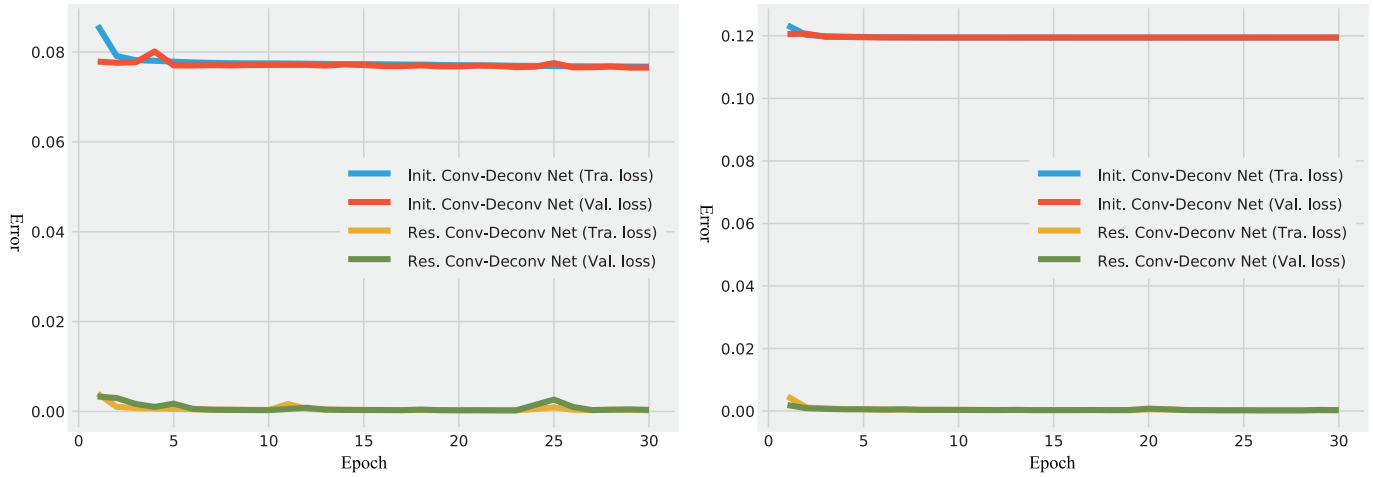


Fig. 9. Learning curves for the initial Conv–Deconv network and the residual Conv–Deconv network on the training samples and the validation samples of (a) Indian Pines data set and (b) Pavia University data set. With residual learning and the new unpooling operation, we can lead the network to a better solution. Here, we use the Adam optimizer with a default learning rate of 0.001.

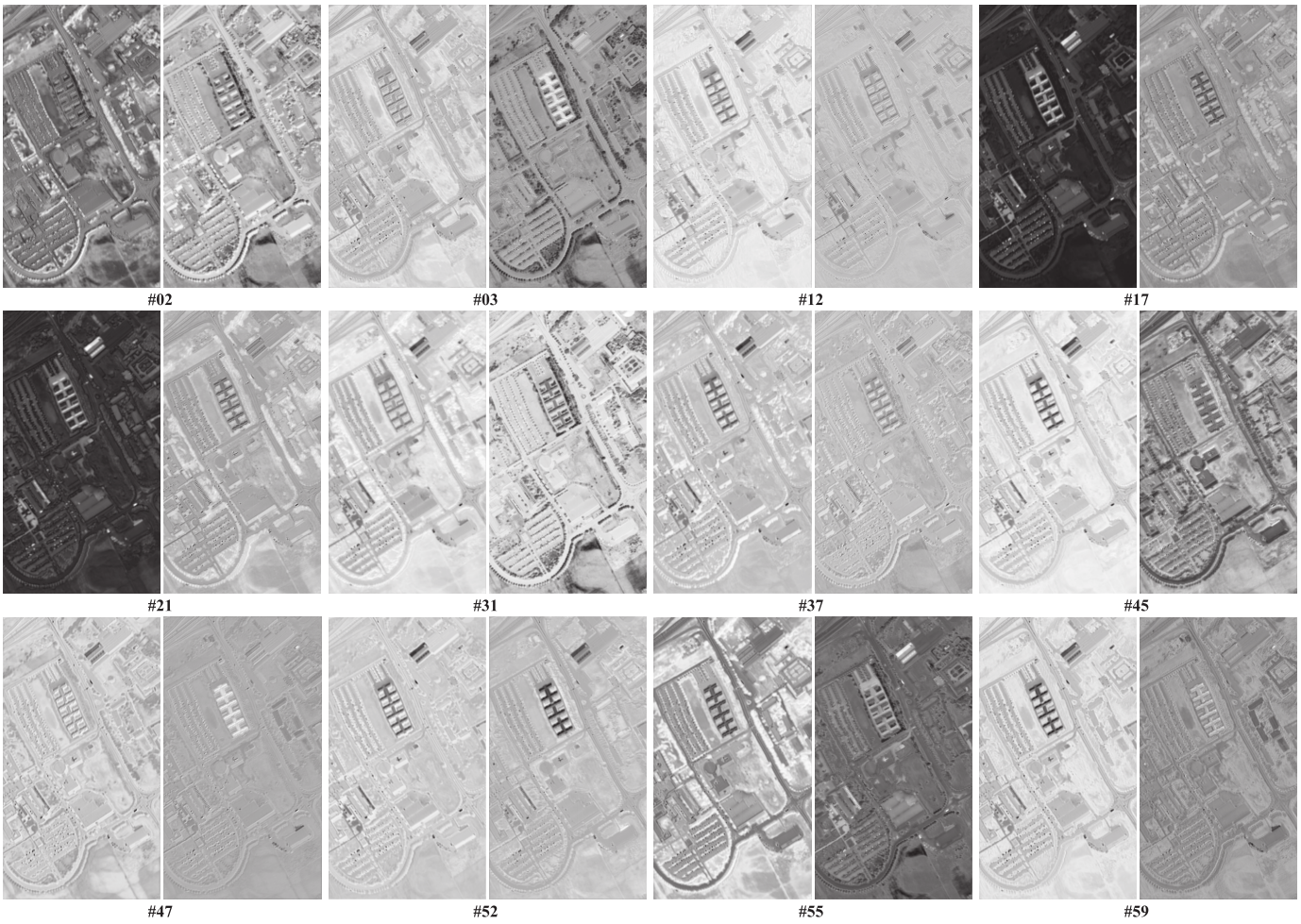


Fig. 10. Feature visualizations from the first residual block of the residual Conv–Deconv network once training is complete on the Pavia University data set. Each group contains two feature maps, including (Left) residual feature $\mathcal{F}(\boldsymbol{\phi}_l; \boldsymbol{\Theta}_l)$ and (Right) output feature map $\boldsymbol{\phi}_{l+1}$. We randomly demonstrate 20 out of 64 learned feature map groups, revealing different structures that are activated by various convolutional filters.

network once training is complete. Each group in Fig. 10 contains two feature maps, i.e., the residual feature $\mathcal{F}(\boldsymbol{\phi}_l; \boldsymbol{\Theta}_l)$ [Fig. 10 (left)] and the output feature $\boldsymbol{\phi}_{l+1}$ [Fig. 10 (right)] of the residual block. We randomly show 20 out of 64 learned feature map groups, revealing the different structures that are activated by various convolutional filters. For instance,

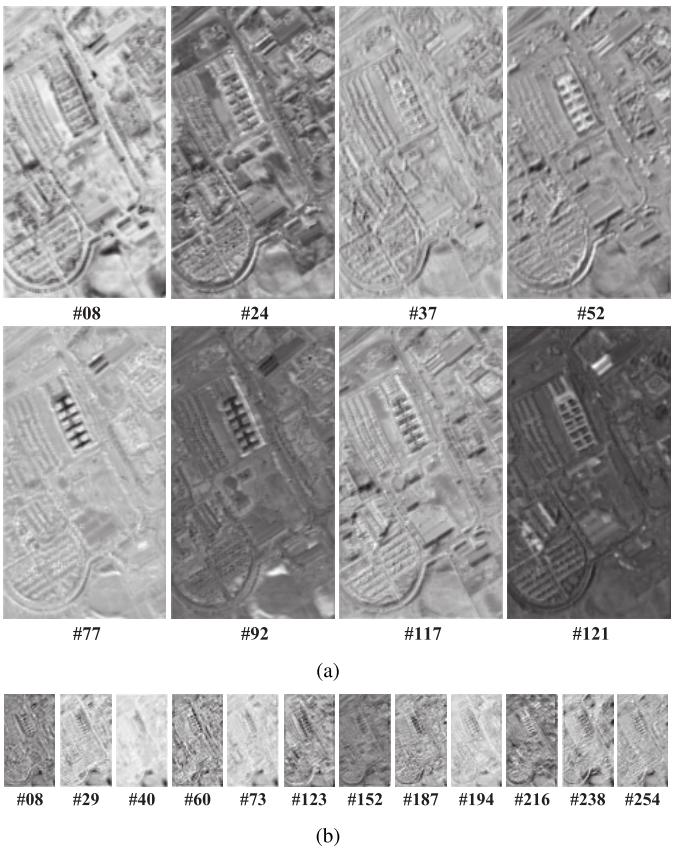


Fig. 11. (a) Eight out of 128 output feature maps of the second residual block. (b) Twelve out of 256 output feature maps of the third residual block.

Fig. 12. Object detection maps of selective convolutional filters from the first residual block of the proposed residual Conv–Deconv network, in which some "neurons" own good description power for semantic visual patterns in the object level. For example, the feature maps activated by the convolutional filters #52 and #03 in the first residual block can be used to precisely capture (a) metal sheets and (b) vegetative covers, respectively. Specifically, we achieve detection by simply setting a global threshold, which is computed by minimizing the intraclass variance of the black and white pixels in the considered feature map [53].

in group #47, the visualization of output feature map reveals that this particular feature focuses on the spectrum of metal sheets in the scene, while the output feature map in group #52 inhibits the expression of the same class. And, as shown in group #37, the residual feature tends to activate the shadow areas in the feature map. Since these feature maps are produced by the corresponding convolutional filters, it is believed that the convolutional filters learned by our residual Conv–Deconv network are capable of extracting some specific spectral–spatial patterns from different perspectives. We also show the output feature maps of the second and the third residual blocks in Fig. 11. It can be seen that the deeper the residual block is, the more abstract the learned feature maps will be naturally.

3) Object Detection: A very interesting thing arises when we analyze the learned feature maps. Although our residual Conv–Deconv network has not been explicitly designed for the task of object detection, we have observed strong evidence of object detection for the hyperspectral image provided by the network at the test stage. In particular, we found that target objects can be localized by the activated or suppressed pixels in some specific learned feature maps of the first residual block. For example, we can determine the objects consisted of metal sheets in the Pavia University data set through finding the hyperspectral pixels that are suppressed by the convolutional filter #52. Similarly, the vegetation covers, including meadows and trees, are able to be identified in the scene by searching the nonactivated pixels in the output feature map #03. To qualitatively assess the object detection results acquired by the proposed approach, examples of such object detection maps are given in Fig. 12. This visualization clearly demonstrates that some "neurons" in the first residual block of the proposed residual Conv–Deconv network know the locations of the target objects within the hyperspectral image and own good description power for semantic visual patterns in the object level. Addressing the detection task seems within reach. Moreover, it is worth noting that compared with the conventional supervised object detectors that need a number of labeled ground truth data, object detection achieved by this method is free and totally unsupervised. Also, as shown in Fig. 12, the quality of such object detection maps is quite good. These maps are with very good edge details, and even very small objects (e.g., cars on the road in the Pavia University scene) can be detected. In a nutshell, our study has shown that the convolutional filters in the proposed residual Conv–Deconv network for the task of unsupervised spectral–spatial feature learning possess strong selectiveness on patterns corresponding to object categories. Particularly, the feature maps obtained by some specific "neurons" at the first residual block of the network record the spectral–spatial representation of visual pattern of a specific object.

D. Fine-Tuned Network for Hyperspectral Image Classification

To further investigate the spectral-spatial features learned by the residual Conv–Deconv network, we evaluated the

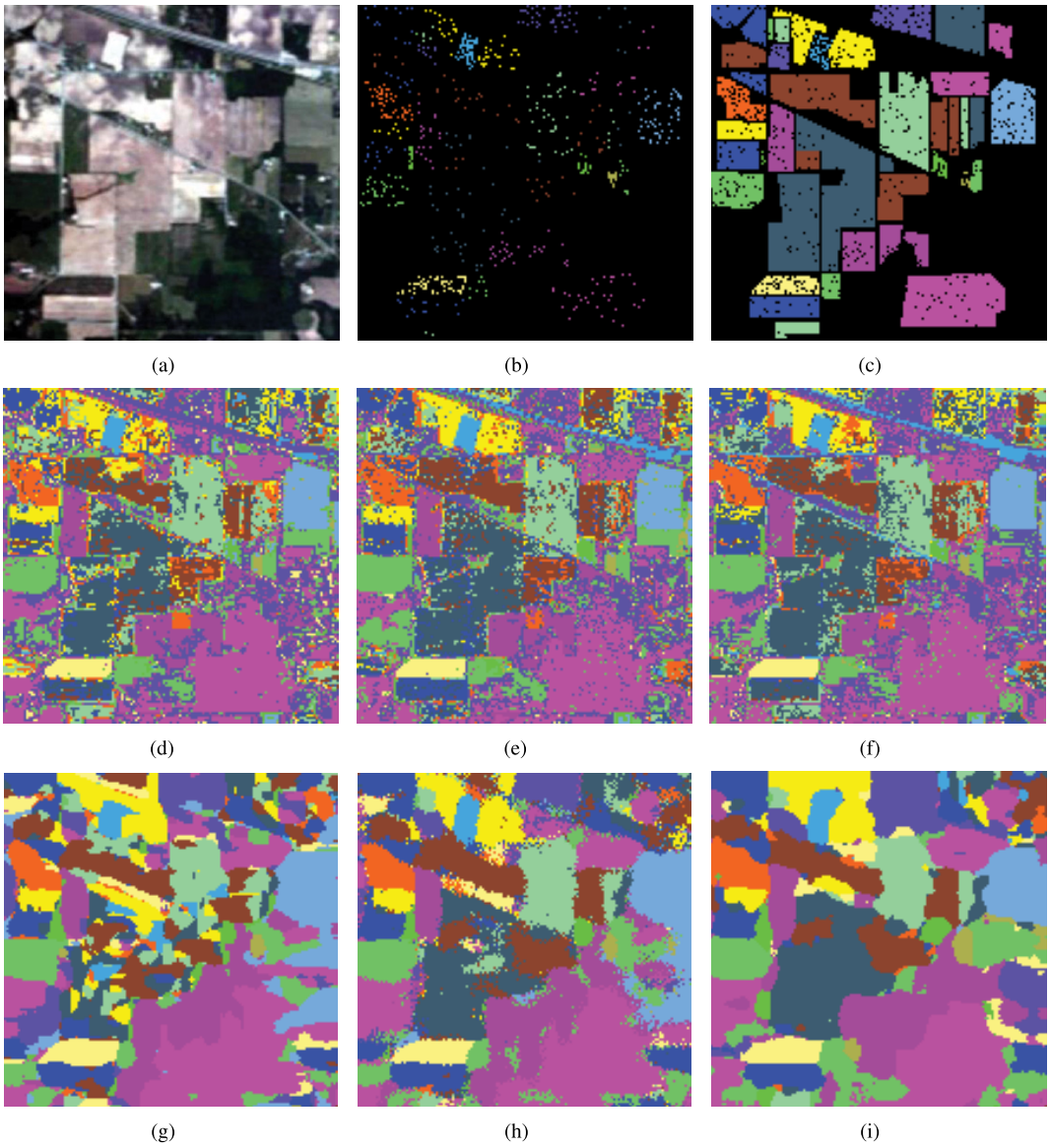


Fig. 13. Classification results obtained by different methods for the Indian Pines scene. (a) True-color composite (bands R: 26, G: 14, B: 8). (b) Training samples. (c) Test samples. (d) RF-200. (e) SVM-RBF. (f) 1-D CNN. (g) 2-D CNN. (h) SICNN. (i) Fine-tuned residual Conv–Deconv network.

performance of the fine-tuned network for the hyperspectral data classification task and provided a comparison with the state-of-the-art approaches.

The classification maps of the Indian Pines data set obtained by the widely used classifiers (e.g., random forest and SVM), supervised CNNs, and our method are shown in Fig. 13, and the corresponding accuracy indexes are presented in Table III. Analysis of the classification accuracy indexes indicates that the SVM with RBF kernel (SVM-RBF) outperforms the random forest classifier, mainly because the kernel SVM generally deals with nonlinear inputs more effectively than the random forest model. The proposed fine-tuned residual Conv–Deconv network achieves better scores for OA and kappa coefficient compared with all other methods. In comparison with SVM-RBF, 1-D CNN, and 2-D CNN, the proposed network increases the OA by 12.98%, 13.36%, and 15.97%, respectively. In addition, the numbers of test samples for different classes of Indian Pines are considerably imbalanced. Hence, the consideration of the OA alone cannot precisely evaluate the usefulness of the classifier, since small classes are commonly ignored. In this case, AA and kappa coefficient can be used to evaluate the performance of different classification models more accurately. Strong difference between the OA and AA or kappa coefficient may means that some classes are incorrectly classified with a high proportion. With respect to these two measures, compared with SVM-RBF, 1-D CNN, and 2-D CNN, the improvements in AA achieved by the proposed network are 9.89%, 12.20%, and 7.58%, respectively, and the increments of kappa coefficient obtained by the fine-tuned residual Conv–Deconv Net are 0.1454, 0.1533, and 0.1406, respectively. Note that the OA and kappa coefficient of 2-D CNN are significantly lower than those of other approaches, as directly training such 2-D network generally suffers from a small and imbalanced data set, while the

TABLE III
ACCURACY COMPARISON FOR THE INDIAN PINES DATA SET. THE BEST ACCURACY IN EACH ROW IS SHOWN IN BOLD

Class No.	Class Name	RF-200	SVM-RBF	1D CNN	2D CNN	SICNN	Res. Conv-Deconv Net
1	Alfalfa	55.71	60.77	56.79	66.98	**79.84**	74.86
2	Corn-notill	58.29	77.68	52.17	80.87	92.47	**95.28**
3	Corn-min	80.98	79.35	85.33	95.65	99.46	**100**
4	Corn	84.79	91.05	87.92	91.95	93.29	**95.08**
5	Grass-pasture	79.77	84.36	85.22	86.94	92.68	**96.56**
6	Grass-trees	95.90	92.03	97.49	97.95	96.58	**99.09**
7	Grass-pasture-mowed	76.58	69.61	74.62	67.86	**86.82**	84.42
8	Hay-windrowed	60.17	59.31	67.99	34.57	69.52	**74.57**
9	Oats	63.12	79.61	58.87	80.85	**83.69**	80.14
10	Soybean-notill	95.68	97.53	98.77	**100**	**100**	**100**
11	Soybean-mintill	88.75	85.21	87.62	88.18	**96.70**	95.74
12	Soybean-clean	53.33	63.64	72.42	91.52	**96.97**	96.06
13	Wheat	97.78	**100**	93.33	**100**	**100**	**100**
14	Woods	56.41	87.18	71.79	71.79	**94.87**	84.62
15	Buildings-grass-trees	81.82	90.91	90.91	**100**	**100**	**100**
16	Stone-steel-towers	**100**	**100**	**100**	**100**	**100**	**100**
OA	-	69.92	72.78	72.40	69.79	85.13	**85.76**
AA	-	76.82	82.39	80.08	84.70	**92.68**	92.28
Kappa	-	0.6605	0.6931	0.6852	0.6979	0.8313	**0.8385**

TABLE IV
CLASSIFICATION ACCURACIES OF DIFFERENT TECHNIQUES IN PERCENTAGE FOR PAVIA UNIVERSITY. THE BEST ACCURACY IN EACH ROW IS SHOWN IN BOLD

Class No.	Class Name	RF-200	SVM-RBF	1D CNN	2D CNN	SICNN	Res. Conv-Deconv Net
1	Asphalt	80.94	84.84	83.73	69.25	**84.21**	78.99
2	Meadows	55.91	67.09	65.70	93.39	91.10	**97.16**
3	Gravel	53.26	**72.13**	67.03	63.13	64.36	61.46
4	Trees	**98.76**	95.72	94.03	94.39	95.53	95.76
5	Metal Sheets	99.11	99.48	99.41	**100**	97.70	97.77
6	Bare Soil	79.26	93.30	**96.30**	49.06	56.53	59.46
7	Bitumen	83.76	91.88	**93.83**	72.26	77.29	79.5
8	Bricks	91.06	92.56	93.56	94.32	95.57	**96.82**
9	Shadows	98.10	97.47	**99.79**	93.77	96.20	92.40
OA	-	71.66	79.88	79.28	82.66	85.25	**87.39**
AA	-	82.24	88.27	**88.15**	81.06	84.28	84.37
Kappa	-	0.6517	0.7487	0.7423	0.7688	0.8041	**0.8308**

proposed strategy, to a large extent, is capable of overcoming this shortcoming. Moreover, SICNN also performs well on the Indian Pines data set, since the specially designed mechanism can effectively solve the curse of dimensionality and the lack of available training samples. But, it is worth noting that our method for feature learning is unsupervised, while 1-D CNN, 2-DCNN, and SICNN are supervised networks. Taking this into account, the performance of our approach is competitive and satisfactory. The proposed approach achieves the best accuracies on most of classes of the Indian Pines data set. For instance, the accuracy of the grass-pasture category obtained by fine-tuned residual Conv–Deconv network reaches 96.56%, and the proposed network can achieve 100% on the corn-min class.

Fig. 14 shows the classification maps using the Pavia University data set; the comparison of accuracies between the random forest, SVM-RBF, supervised CNNs, and our approach can be found in Table IV. It can be seen that the proposed fine-tuned residual Conv–Deconv network outperforms the others in terms of OA and kappa coefficient. Misclassification in this data set lies in similar objects, such as Meadow-Trees. The proposed network achieves the best AA of 96.46% on Meadow-Trees. Similarly, the misclassification problem in the Indian pines data set is also improved. For example, the AA of Corn-notill, Corn-min, and Corn obtained by the fine-tuned residual Conv–Deconv network is 96.79%, which is higher than that of SVM-RBF (82.69%), 1-D CNN (75.14%), 2-D CNN (89.49%), and SICNN (95.07%). Furthermore, in Figs. 13 and 14, it is obvious that the spectral classification methods (random forest, SVM, and 1-D CNN) always result in noisy scatter points in the classification maps, while the spectral–spatial approaches (2-D CNN, SICNN, and fine-tuned residual Conv–Deconv network) address this problem by eliminating noisy scattered points of misclassification.

In addition to comparing the proposed approach with the traditional classifiers (random forest and SVM) and other deep networks, some mathematical morphology-based methods like the morphological profile (MP) [54] are also considered in comparison due to their capacity to extract spatial features. Fauvel *et al.* [55] summarized some frequently used spectral–spatial features. Benediktsson *et al.* [56] proposed an extended MP (EMP) using principal component analysis (PCA) for hyperspectral image classification. The EMP-PCA [56] is able to achieve the OA of 77.7%, AA of 82.5%, and kappa coefficient of 0.71 on the Pavia University

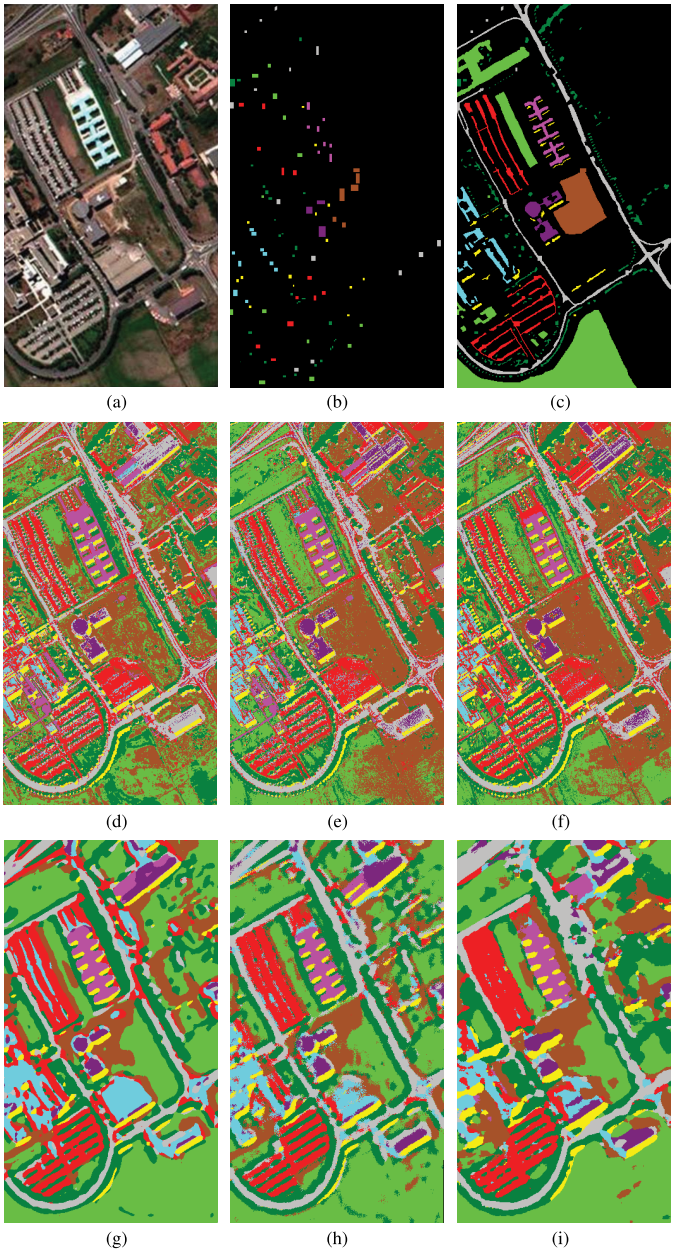


Fig. 14. Classification results obtained by different methods for the Pavia University scene. (a) Composite image of hyperspectral data. (b) Training data. (c) Ground truth reference. (d) RF-200. (e) SVM-RBF. (f) 1-D CNN. (g) 2-D CNN. (h) SICNN. (i) Fine-tuned residual Conv–Deconv network.

data set. Fauvel *et al.* [57] attempted to make use of kernel PCA to produce EMP, in which state-of-the-art performance on the Pavia University scene can be obtained with the OA of 96.3%, AA of 95.7%, and kappa coefficient of 0.95. For more mathematical morphology-based approaches, please refer to [55].

Table V gives information about the results of McNemar's test to evaluate the significance of the difference between the classification accuracies of the proposed network and the other investigated approaches. With reference to Table V, the improvements of OAs achieved by the proposed methods are statistically significant in comparison with the other studied methods. It is worth noting that the SICNN performs similarly to the proposed approach on the Indian Pines data set

TABLE V

ASSESSMENT OF THE SIGNIFICANCE OF THE CLASSIFICATION ACCURACIES OF THE PROPOSED METHOD COMPARED WITH THE OTHER INVESTIGATED APPROACHES FOR BOTH THE INDIAN PINES AND PAVIA UNIVERSITY DATA SETS

Data set	RF-200	SVM-RBF	1D CNN	2D CNN	SICNN
Indian Pines	28.056	24.995	24.440	32.463	1.747
Pavia University	56.949	29.128	31.362	31.464	12.029

TABLE VI

STATISTICS OF TRAINING TIME (MINUTES)

Data set	Res. Conv-Deconv Net	Fine-tuned network
Indian Pines	20.3	3.1
Pavia University	34.8	6.9

(the value is 1.747), as the SICNN exploits band selection before feeding the data into the CNN, which greatly reduces the total number of parameters of the network and thus improves the accuracy.

E. Processing Time

For both training and testing steps of the residual Conv–Deconv network and the fine-tuned network, we have used an NVIDIA GTX Titan GPU. The other approaches, i.e., random forest, SVM-RBF, and 1-D CNN, are computed on a CPU with a personal computer equipped with an Intel Core I5 with 2.20 GHz. The training times of the residual Conv–Deconv network and the fine-tuned network are shown in Table VI. With the help of GPU, the training times of the proposed networks are acceptable.

V. CONCLUSION

In this paper, we proposed a novel end-to-end fully Conv–Deconv network architecture for unsupervised spectral–spatial feature extraction of hyperspectral images. In particular, the proposed network is composed of two parts, namely, the convolutional subnetwork and deconvolutional subnetwork. They are responsible for transforming an input 3-D hyperspectral patch to abstract feature representation and reproducing the initial input data from the encoded feature, respectively. Furthermore, residual learning and a new unpooling operation that can make use of max-pooling indexes are introduced to our network architecture in order to overcome the training problem caused by vanishing gradient. A very interesting observation can be found when we analyze the learned feature maps. Although the proposed network has not been explicitly designed for the task of object detection, we have observed that target object can be localized by the activated or suppressed pixels in some specific learned feature maps of the first residual block, which makes it possible to achieve the unsupervised object detection in hyperspectral images. Experimental results also demonstrate that the features learned by the proposed unsupervised network can be used for the hyperspectral image classification task, and the obtained classification results are competitive compared with the other supervised approaches.

In the future, further experiments and studies will be conducted to fully understand the "block box" of the proposed

fully Conv–Deconv network with residual learning, providing more accurate analysis for remote sensing applications such as unsupervised object detection with the help of learned feature maps. In addition, the input to the proposed Conv–Deconv network is the raw hyperspectral data, and a possible future work is to explore the capability of the proposed approach using APs and extinction profiles that extract spatial information in a robust and adaptive way.

ACKNOWLEDGMENT

The authors would like to thank Prof. Paolo Gamba from the University of Pavia, Italy, for providing the ROSIS data and corresponding reference information and Prof. Landgrebe from Purdue University for providing the Indian Pines data set.

REFERENCES

[1] J. A. Benediktsson and P. Ghamisi, *Spectral-Spatial Classification of Hyperspectral Remote Sensing Images*. Boston, MA, USA: Artech House, 2015.

[2] P. Ghamisi, M. Dalla Mura, and J. A. Benediktsson, "A survey on spectral–spatial classification techniques based on attribute profiles," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 5, pp. 2335–2353, May 2015.

[3] Y. Gu, T. Liu, X. Jia, J. A. Benediktsson, and J. Chanussot, "Nonlinear multiple kernel learning with multiple-structure-element extended morphological profiles for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 6, pp. 3235–3247, Jun. 2016.

[4] J. Li, M. Khodadadzadeh, A. Plaza, X. Jia, and J. M. Bioucas-Dias, "A discontinuity preserving relaxation scheme for spectral–spatial hyperspectral image classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 9, no. 2, pp. 625–639, Feb. 2016.

[5] P. Ghamisi, J. A. Benediktsson, and J. R. Sveinsson, "Automatic spectral–spatial classification framework based on attribute profiles and supervised feature extraction," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 9, pp. 5771–5782, Sep. 2014.

[6] C. Wu, B. Du, and L. Zhang, "Slow feature analysis for change detection in multispectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 5, pp. 2858–2874, May 2014.

[7] H. Lyu, H. Lu, and L. Mou, "Learning a transferable change rule from a recurrent neural network for land cover change detection," *Remote Sens.*, vol. 8, no. 6, p. 506, 2016.

[8] B. Demir, F. Bovolo, and L. Bruzzone, "Updating land-cover maps by classification of image time series: A novel change-detection-driven transfer learning approach," *IEEE Trans. Geosci. Remote Sens.*, vol. 51, no. 1, pp. 300–312, Jan. 2013.

[9] J. Meola, M. T. Eismann, R. L. Moses, and J. N. Ash, "Application of model-based change detection to airborne VNIR/SWIR hyperspectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 10, pp. 3693–3706, Oct. 2012.

[10] L. G. Olmanson, P. L. Brezonik, and M. E. Bauer, "Airborne hyperspectral remote sensing to assess spatial distribution of water quality characteristics in large rivers: The Mississippi River and its tributaries in Minnesota," *Remote Sens. Environ.*, vol. 130, pp. 254–265, Mar. 2013.

[11] M. S. Moran, Y. Inoue, and E. M. Barnes, "Opportunities and limitations for image-based remote sensing in precision crop management," *Remote Sens. Environ.*, vol. 61, no. 3, pp. 319–346, Sep. 1997.

[12] S. Delalieux, B. Somers, B. Haest, T. Spanhove, J. V. Borre, and C. A. Mücher, "Heathland conservation status mapping through integration of hyperspectral mixture analysis and decision tree classifiers," *Remote Sens. Environ.*, vol. 126, pp. 222–231, Nov. 2012.

[13] J. Ham, Y. Chen, M. M. Crawford, and J. Ghosh, "Investigation of the random forest framework for classification of hyperspectral data," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 3, pp. 492–501, Mar. 2005.

[14] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 8, pp. 832–844, Aug. 1998.

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

[16] J. A. Gualtieri and S. Chettri, "Support vector machines for classification of hyperspectral data," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2000, pp. 813–815.

[17] J. Li, J. M. Bioucas-Dias, and A. Plaza, "Spectral–spatial classification of hyperspectral data using loopy belief propagation and active learning," *IEEE Trans. Geosci. Remote Sens.*, vol. 51, no. 2, pp. 844–856, Feb. 2013.

[18] M. Fauvel, J. A. Benediktsson, J. Chanussot, and J. R. Sveinsson, "Spectral and spatial classification of hyperspectral data using SVMs and morphological profiles," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 11, pp. 3804–3814, Nov. 2008.

[19] P. Ghamisi, Y. Chen, and X. X. Zhu, "A self-improving convolution neural network for the classification of hyperspectral data," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 10, pp. 1537–1541, Oct. 2016.

[20] Y. Chen, Z. Lin, X. Zhao, G. Wang, and Y. Gu, "Deep learning-based classification of hyperspectral data," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 6, pp. 2094–2107, Jun. 2014.

[21] Y. Chen, X. Zhao, and X. Jia, "Spectral–spatial classification of hyperspectral data based on deep belief network," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 8, no. 6, pp. 2381–2392, Jun. 2015.

[22] C. Tao, H. Pan, Y. Li, and Z. Zou, "Unsupervised spectral–spatial feature learning with stacked sparse autoencoder for hyperspectral imagery classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 12, pp. 2438–2442, Dec. 2015.

[23] L. Mou, P. Ghamisi, and X. X. Zhu, "Deep recurrent neural networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 7, pp. 3639–3655, Jul. 2017.

[24] P. Ghamisi and J. A. Benediktsson, "Feature selection based on hybridization of genetic algorithm and particle swarm optimization," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 2, pp. 309–313, Feb. 2015.

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.

[26] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. IEEE Int. Conf. Learn. Represent. (ICLR)*, May 2015.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[28] J. Hu, L. Mou, A. Schmitt, and X. X. Zhu, "FusioNet: A two-stream convolutional neural network for urban scene classification using PolSAR and hyperspectral data," in *Proc. Joint Urban Remote Sens. Event (JURSE)*, 2017, pp. 1–4.

[29] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 1, pp. 142–158, Jan. 2016.

[30] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[31] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.

[32] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1520–1528.

[33] Y. Yuan, L. Mou, and X. Lu, "Scene recognition by manifold regularized deep learning architecture," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 10, pp. 2222–2233, Oct. 2015.

[34] L. Mou and X. X. Zhu, "Spatiotemporal scene interpretation of space videos via deep neural network and tracklet analysis," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2016, pp. 1823–1826.

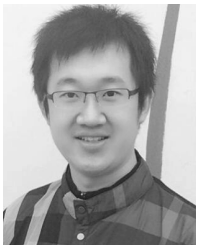
[35] L. Mou *et al.*, "Multitemporal very high resolution from space: Outcome of the 2016 IEEE GRSS Data Fusion Contest," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 10, no. 8, pp. 3435–3447, Aug. 2017.

[36] Y. Chen, H. Jiang, C. Li, X. Jia, and P. Ghamisi, "Deep feature extraction and classification of hyperspectral images based on convolutional neural networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 10, pp. 6232–6251, Oct. 2016.

[37] K. Makantasis, K. Karantzalos, A. Doulamis, and N. Doulamis, "Deep supervised learning for hyperspectral data classification through convolutional neural networks," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2015, pp. 4959–4962.

[38] W. Zhao and S. Du, "Spectral–spatial feature extraction for hyperspectral image classification: A dimension reduction and deep learning approach," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 8, pp. 4544–4554, Aug. 2016.

[39] E. Aptoula, M. C. Ozdemir, and B. Yanikoglu, "Deep learning with attribute profiles for hyperspectral image classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 12, pp. 1970–1974, Dec. 2016.
[40] A. Romero, C. Gatta, and G. Camps-Valls, "Unsupervised deep feature extraction for remote sensing image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 3, pp. 1349–1362, Mar. 2016.
[41] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, no. 12, pp. 3371–3408, Dec. 2010.
[42] Y. LeCun *et al.*, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, 1989.
[43] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.
[44] A. Dosovitskiy, J. T. Springenberg, and T. Brox, "Learning to generate chairs, tables and cars with convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1538–1546.
[45] A. Dosovitskiy, P. Fischer, J. T. Springenberg, M. Riedmiller, and T. Brox, "Discriminative unsupervised feature learning with exemplar convolutional neural networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 9, pp. 1734–1747, Sep. 2016.
[46] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.
[47] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. Int. Conf. Artif. Intell. Statist. (AISTATS)*, 2010, pp. 249–256.
[48] R. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," in *Proc. IEEE Int. Conf. Mach. Learn. (ICML)*, Jul. 2015.
[49] R. Goroshin, M. F. Mathieu, and Y. LeCun, "Learning to linearize under uncertainty," in *Proc. Annu. Conf. Neural Inf. Process. Syst. (NIPS)*, 2015, pp. 1234–1242.
[50] G. M. Foody, "Thematic map comparison: Evaluating the statistical significance of differences in classification accuracy," *Photogramm. Eng. Remote Sens.*, vol. 70, no. 5, pp. 627–633, 2004.
[51] W. Hu, Y. Huang, L. Wei, F. Zhang, and H. Li, "Deep convolutional neural networks for hyperspectral image classification," *J. Sensors*, vol. 2015, Jan. 2015, Art. no. 258619.
[52] D. P. Kingma and J. Ba. (2015). "Adam: A method for stochastic optimization." [Online]. Available: https://arxiv.org/abs/1412.6980
[53] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-9, no. 1, pp. 62–66, Jan. 1979.
[54] M. Pesaresi and J. A. Benediktsson, "A new approach for the morphological segmentation of high-resolution satellite imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 39, no. 2, pp. 309–320, Feb. 2001.
[55] M. Fauvel, Y. Tarabalka, J. A. Benediktsson, J. Chanussot, and J. C. Tilton, "Advances in spectral-spatial classification of hyperspectral images," *Proc. IEEE*, vol. 101, no. 3, pp. 652–675, Mar. 2013.
[56] J. A. Benediktsson, J. A. Palmason, and J. R. Sveinsson, "Classification of hyperspectral data from urban areas based on extended morphological profiles," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 3, pp. 480–491, Mar. 2005.
[57] M. Fauvel, J. Chanussot, and J. A. Benediktsson, "Kernel principal component analysis for the classification of hyperspectral remote sensing data over urban areas," *EURASIP J. Adv. Signal Process.*, vol. 2009, p. 783194, Dec. 2009.



Lichao Mou (S'16) received the bachelor's degree in automation from the Xi'an University of Posts and Telecommunications, Xi'an, China, in 2012, and the master's degree in signal and information processing from the University of Chinese Academy of Sciences, Beijing, China, in 2015. He is currently pursuing the Ph.D. degree with the German Aerospace Center, Wessling, Germany, and the Technical University of Munich, Munich, Germany.

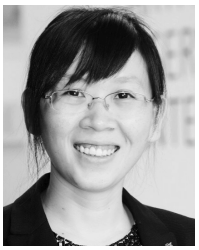
In 2015, he joined the Computer Vision Group, University of Freiburg, Freiburg im Breisgau, Germany. His research interests include remote sensing, computer vision, and machine learning, especially remote sensing video analysis and deep networks with their applications in remote sensing.

Mr. Mou won the first place prize in the 2016 IEEE GRSS Data Fusion Contest.

Pedram Ghamisi (S'12–M'15) received the B.Sc. degree in civil (survey) engineering from the Tehran South Campus of Azad University, Tehran, Iran, the M.Sc. (First Class Hons.) degree in remote sensing from the K. N. Toosi University of Technology, Tehran, in 2012, and the Ph.D. degree in electrical and computer engineering from the University of Iceland, Reykjavik, Iceland, in 2015.

In 2013 and 2014, he joined the School of Geography, Planning and Environmental Management, University of Queensland, Brisbane, QLD, Australia. He was a Post-Doctoral Research Fellow with the University of Iceland. He has been a Post-Doctoral Research Fellow with the Technical University of Munich, Munich, Germany, and Heidelberg University, Heidelberg, Germany, since 2015. He has also been a Researcher with the German Aerospace Center, Remote Sensing Technology Institute, Wessling, Germany, since 2015. His research interests include remote sensing and image analysis, with a special focus on the spectral and spatial techniques for hyperspectral image classification, multisensor data fusion, machine learning, and deep learning.

Dr. Ghamisi received the Best Researcher Award for M.Sc. students from the K. N. Toosi University of Technology. In 2013, he presented at the IEEE International Geoscience and Remote Sensing Symposium, Melbourne, VIC, Australia, and was awarded the IEEE Mikio Takagi Prize for winning the conference Student Paper Competition against almost 70 people. In 2016, he was selected as a Talented International Researcher by the Iran's National Elites Foundation. In 2017, he won the Data Fusion Contest 2017 organized by the Image Analysis and Data Fusion Technical Committee of the Geoscience and Remote Sensing Society. His model was the most accurate among more than 800 submissions. He received the prestigious Alexander von Humboldt Fellowship in 2015.



Xiao Xiang Zhu (S'10–M'12–SM'14) received the bachelor's degree in space engineering from the National University of Defense Technology, Changsha, China, in 2006, and the M.Sc., Dr.-Ing., and "Habilitation" degrees in signal processing from the Technical University of Munich (TUM), Munich, Germany, in 2008, 2011, and 2013, respectively.

She was a Guest Scientist or a Visiting Professor with the Italian National Research Council, Naples, Italy, in 2009; Fudan University, Shanghai, China, in 2014; the University of Tokyo, Tokyo, Japan, in 2015; and the University of California, Los Angeles, CA, USA, in 2016. She has been the Professor of signal processing in earth observation with TUM since 2015; the Head of the Team Signal Analysis, German Aerospace Center (DLR), Remote Sensing Technology Institute, since 2011; and the Head of the Helmholtz Young Investigator Group "SiPEO," DLR, and TUM, since 2013. Her research interests include advanced InSAR techniques such as high-dimensional tomographic SAR imaging and SqueeSAR; computer vision in remote sensing including object reconstruction and multidimensional data visualization; and big data analysis in remote sensing and modern signal processing, including innovative algorithms such as sparse reconstruction, nonlocal means filter, robust estimation, and deep learning, with applications in the field of remote sensing such as multi/hyperspectral image analysis.

Dr. Zhu is a member of young academy (Junge Akademie/Junges Kolleg) at the Berlin-Brandenburg Academy of Sciences and Humanities and the German National Academy of Sciences Leopoldina and the Bavarian Academy of Sciences and Humanities. He is an Associate Editor of the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING.