Deep Learning-Based Classification of Hyperspectral Data

Yushi Chen, *Member, IEEE,* Zhouhan Lin, Xing Zhao, *Student Member, IEEE,* Gang Wang, *Member, IEEE,* and Yanfeng Gu, *Member, IEEE*

***Abstract*—Classification is one of the most popular topics in hyperspectral remote sensing. In the last two decades, a huge number of methods were proposed to deal with the hyperspectral data classification problem. However, most of them do not hierarchically extract deep features. In this paper, the concept of deep learning is introduced into hyperspectral data classification for the first time. First, we verify the eligibility of stacked autoencoders by following classical spectral information-based classification. Second, a new way of classifying with spatial-dominated information is proposed. We then propose a novel deep learning framework to merge the two features, from which we can get the highest classification accuracy. The framework is a hybrid of principle component analysis (PCA), deep learning architecture, and logistic regression. Specifically, as a deep learning architecture, stacked autoencoders are aimed to get useful high-level features. Experimental results with widely-used hyperspectral data indicate that classifiers built in this deep learning-based framework provide competitive performance. In addition, the proposed joint spectral–spatial deep neural network opens a new window for future research, showcasing the deep learning-based methods' huge potential for accurate hyperspectral data classification.**

***Index Terms*—Autoencoder (AE), deep learning, feature extraction, hyperspectral data classification, logistic regression, stacked autoencoder (SAE), support vector machine (SVM).**

I. Introduction

BY COMBINING imaging and spectroscopy technology, hyperspectral remote sensing can get spatially and spectrally continuous data simultaneously. Hyperspectral data are becoming a valuable tool for monitoring the Earth's surface [1], [2], and are used in a wide array of applications. An incomplete list includes agriculture [3], mineralogy [4], surveillance [5], physics [6], astronomy [7], chemical imaging [8], and environmental sciences [9], [10]. A common technique in these applications is the classification of each pixel in hyperspectral data. If successfully exploited, the hyperspectral data can yield higher classification accuracies and more detailed class taxonomies [11]. However, there are several critical problems in the classification of hyperspectral data: 1) curse of dimensionality, because of the high number of spectral channels; 2) limited number of labeled training samples; and 3) large spatial variability of spectral signature [12].

A lot of different classification methods have been proposed to deal with hyperspectral data classification. Traditional hyperspectral data classification methods use spectral information only, and the classification algorithms typically include parallelepiped classification, k-nearest-neighbors, maximum-likelihood, minimum distance, and logistic regression [13]. The majority of these above algorithms suffer a lot from the "curse of dimensionality." To deal with the high dimensionality and limited training samples of hyperspectral data [14], some dimensionality reduction-based classification methods were proposed. Transformation is one method available to deal with high dimensionality [15]–[17]. Band selection is another method available to mitigate this "curse" [18]–[20].

In [21], a promising classification method, support vector machine (SVM), is introduced for hyperspectral data classification. SVM exhibits low sensitivity to high dimensionality and is unlikely to suffer from the Hughes phenomenon [22]. In most cases, SVM-based classifiers can obtain better classification accuracy than other widely used pattern recognition techniques [14], [22]. For a long time, these classifiers were the state-of-the-art methods [23].

Spatial information has been growing more and more important for hyperspectral data classification in recent years [30]. Spatial–spectral classification methods provide significant advantages in terms of improving performance [10]. To deal with spatial variability of spectral signature, some recent approaches try to incorporate spatial information into consideration [31]–[33]. In [32], the proposed method based on the fusion of morphological information and original data followed by SVM provides good classification results. In [33], a new classification framework is proposed to exploit the spatial and spectral information using loopy belief propagation and active learning. In recent years, sparse representation-based methods have been widely used in many fields. In [34], spatial–spectral kernel sparse representation is proposed to deal with hyperspectral data classification.

Considering the machine learning task of classification, classifiers like linear SVM and logistic regression can be attributed to single-layer classifiers, whereas decision tree or SVM with kernels are believed to have two layers [24]. As is confirmed in neuroscience, human brains perform well in tasks like object recognition because of its multiple stages of processing from

Manuscript received October 14, 2013; revised April 24, 2014; accepted May 30, 2014. Date of publication June 25, 2014; date of current version August 01, 2014. This work was supported in part by the Fundamental Research Funds for the Central Universities under Grant HIT. NSRIF.2013028 and in part by National Natural Science Foundation of China under Grant 61301206 and Grant 61371180. *(Corresponding author: Yushi Chen.)*

Y. Chen, Z. Lin, X. Zhao, and Y. Gu are with the Institute of Image and Information Technology, Harbin Institute of Technology, Harbin 150001, China (e-mail: chenyushi@hit.edu.cn; lin.zhouhan@gmail.com; zhaoxing@hit.edu.cn; yfgu@hit.edu.cn).

G. Wang is with the School Electrics and Electronics Engineering, Nanyang Technological University, 639798 Singapore (e-mail: wanggang@ntu.edu.sg).

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/JSTARS.2014.2329330

retina to cortex [25]. Similarly, machine learning systems with multiple layers of processing extract more abstract, invariant features of data, and thus are believed to have the ability of yielding higher classification accuracy than those traditional, shallower classifiers. These deep architectures have been shown to yield promising performance in many field including classification or regression tasks that involve image [26], [27], [47], language [28], and speech [29].

In this paper, we introduce deep learning-based feature extraction for hyperspectral data classification for the first time. Our work focuses on applying autoencoder (AE), which is one of the deep architecture-based models, to learn deep features of hyperspectral data in an unsupervised manner. Our methods exploit single-layer AE and multi-layer stacked AE (SAE) to learn shallow and deep features of hyperspectral data, respectively. Furthermore, we propose a new way of extracting spatial-dominated information for classification. At last, we propose a novel classification framework dealing with joint spectral–spatial information, which utilizes all of the features extracted in the former two sections.

The rest of this paper is organized into six sections. Section II is a description of deep learning, AE, and SAE models used in this paper. In Section III, we focus on classifying with spectral features, whereas Section IV details a new way of incorporating spatial information by extracting spatial-dominated features. In Section V, we further merge the former two spectral and spatial approaches and propose a novel joint spectral–spatial deep learning framework, which yields the highest classification accuracy. Experimental results are shown in Section VI. Section VII summarizes the observations and completes this paper.

II. Deep Learning, AE, and SAE

A. Deep Learning

As early as 1989, the universal expressive power of three-layer nets was proved via bumps and Fourier ideas [35]. The proof showed surprisingly that any continuous function from input to output can be implemented in a three-layer net, given sufficient number of hidden units and proper nonlinearities in activation function and weights. However, due to the lack of proper training algorithms in early years, people could not harness this powerful model until Hinton proposed his deep learning idea in 2006 [27].

Deep learning involves a class of models which try to hierarchically learn deep features of input data with very deep neural networks, typically deeper than three layers. The network is first layer-wise initialized via unsupervised training and then tuned in a supervised manner. In this scheme, high-level features can be learned from low-level ones, whereas the proper features can be formulated for pattern classification in the end. Deep models can potentially lead to progressively more abstract and complex features at higher layers, and more abstract features are generally invariant to most local changes of the input. According to some recent papers [36], [37], deep models can give better approximation to nonlinear functions than shallow models.

Typical deep neural network architectures include deep belief networks (DBNs) [38], deep Boltzmann machines (DBMs) [39], SAEs [40], and stacked denoising AEs (SDAEs) [41].

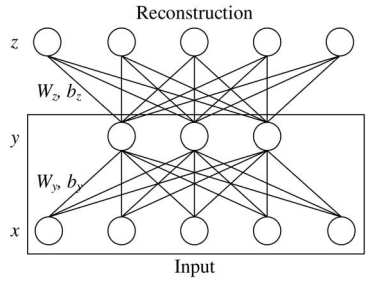


Fig. 1. Single layer AE for hyperspectral data classification. The model learns a hidden feature " y" from input " x" by reconstructing it on " z." Corresponding parameters are denoted in the network.

The layer-wise training models have a bunch of alternatives such as restricted Boltzmann machines (RBMs) [42], pooling units [43], convolutional neural networks (CNNs) [44], AEs, and denoising AEs (DAE) [40]. In this paper, we adopt one of the above deep learning models, AE, for hyperspectral data classification and choose SAEs as the corresponding deep architecture.

B. Autoencoders

An AE has one visible layer of d inputs, one hidden layer of h units, one reconstruction layer of d units, and an activation function f (Fig. 1).

During training, it first maps the input $x \epsilon R^d$ to the hidden layer and produces the latent activity $y \epsilon R^h$. The network corresponding to this step is shown in the boxed part of Fig. 1 and is called an "encoder." Then, y is mapped by a "decoder" to an output layer that has the same size of the input layer, which is called "reconstruction." The reconstructed values are denoted as $z \epsilon R^d$. Mathematically, these two steps can be formulated as

$$y = f(W_y x + b_y) \tag{1}$$

$$z = f(W_z y + b_z) \tag{2}$$

where W_y and W_z denote the input-to-hidden and the hidden-to-output weights, respectively, b_y and b_z denote the bias of hidden and output units, and $f(\cdot)$ denotes the activation function. Conventionally, the nonlinearity is provided in $f(\cdot)$. There are a lot of alternatives for $f(\cdot)$ such as sigmoid function, hyperbolic tangent, and rectified linear function.

In our paper, the following constraint holds

$$W_y = W'_z = W. \tag{3}$$

We say that the AE has *tied weights*, which helps to halve model parameters. Thus, we have three groups of parameters remaining to learn: W, b_y, b_z.

The goal of training is to minimize the "error" between input and reconstruction, i.e.,

$$\underset{W,b_y,b_z}{argmin}[c(x,z)] \tag{4}$$

where z is dependent on parameters W, b_y, b_z while x is given. $c(x, z)$ stands for the "error," which can be defined in a variety of

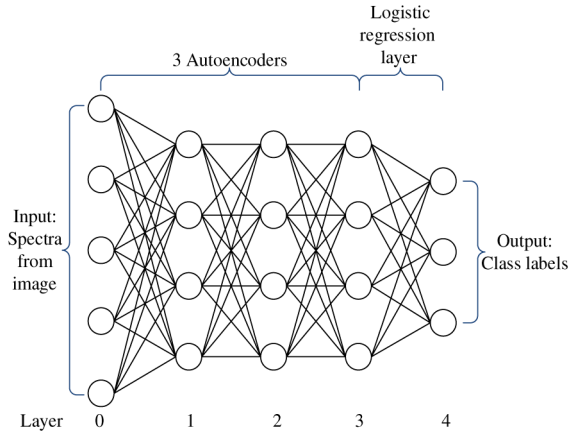


Fig. 2. Instance of a SAE connected with a logistic regression layer. It has five layers: one input layer, three hidden layers, and an output layer.

ways. Thus, the weight updating rule can be defined as (where η denotes the learning rate)

$$W = W - \eta \frac{\partial cost(x,z)}{\partial W} \tag{5}$$

$$b_y = b_y - \eta \frac{\partial cost(x,z)}{\partial b_y} \tag{6}$$

$$b_z = b_z - \eta \frac{\partial cost(x,z)}{\partial b_z}. \tag{7}$$

After training the network, the reconstruction layer together with its parameters are removed and the learned feature lies in the hidden layer, which can subsequently be used for classification or used as the input of a higher layer to produce a deeper feature.

The power of AE lies in this form of reconstruction-oriented training. Note that during reconstruction, it only uses the information in hidden layer activity y, which is encoded as features from input. If the model can recover original input perfectly from y, it means that y retains enough information of the input. And the learned nonlinear transformation, which is defined by those weights and biases, can be deemed as a good feature extraction step. So, stacking the encoders trained in this manner minimizes information loss. At the meantime, they preserve abstract and invariant information in deeper feature. This is the reason why we choose AE to progressively extract deep features for hyperspectral data.

C. Stacked AE

Stacking the input and hidden layers of AEs together layer by layer constructs a SAE. The model is used to generate deep features of hyperspectral data. Fig. 2 shows a typical instance of a SAE connected with a subsequent logistic regression classifier.

The first AE maps inputs in 0th layer to a first layer feature in first layer. It is trained using the same method introduced in Section II-B. After we finish training the first layer AE, subsequent layers of AEs are trained via the output of its previous layer. For example, although we are training the AE between the second and third layer, we try to reconstruct the output of the second layer according to the activity of the third layer. After this layer of training, the decoder of the third layer AE is cast away and only the input-to-hidden parameters are incorporated as weights between the second and the third layer.

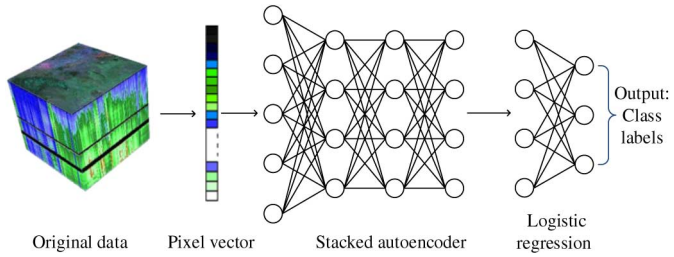


Fig. 3. Classifying with spectral feature. The classification scheme shown here has five layers: one input layer, three hidden layers of AEs, and an output layer of logistic regression. If we want to learn a shallower feature set, we just remove the higher layers of AE.

If the subsequent classifier is implemented as a neural network too, parameters throughout the whole network can be adjusted slightly while we are training the classifier. This step is called fine-tuning. For logistic regression, the training is simply back propagation, searching for a minimum in a peripheral region of parameters initialized by the former step.

III. Classifying With Spectral Features

There exist some motivations to extract robust deep spectral features. First, because of the complex situation of lighting in the large scene, objects of the same class show different spectral characteristics in different locations. For example, a lawn exposed to direct sunlight shows different spectral characteristics from a similar lawn eclipsed from the sunlight by a high building. Also, scattering from other peripheral ground objects tilts the spectra of the lawn and changes its characteristics too. Other factors involve rotations of the sensor, different atmospheric scattering conditions, and so on. According to these factors, the probability distribution of a certain class is hard to be one-hot and has variations over multiple directions in the feature space. These complex variations of spectra make it hopeless to analyze pixel by pixel how they are affected by their tangent pixels in the complicated real situation, thus they demand more robust and invariant features. It is believed that deep architectures can potentially lead to progressively more abstract features at higher layers of feature, and more abstract features are generally invariant to most local changes of the input [24].

To get more generally invariant features and tackle these problems, a deep spectral feature of hyperspectral data can be learned progressively layer by layer with the aforementioned AE models. Generally speaking, we first compute features via a SAE and deem them as the features of data, then construct a logistic regression classifier on top of the neural network to finish the classification phase. By adjusting different numbers of layers of AEs, both shallow and deep features can be learned. Fig. 3 shows a typical instance of the deep architecture used in our paper. The training procedure will be detailed below.

A. Hierarchal Pretraining

The first stage is to learn a deep feature of spectra via pretraining a SAE in a hierarchal manner, which is outlined in

Section II-C. Here, we derive the detailed training criterion while training an AE within each layer.

To get a nonlinear mapping, the activation function $f(\cdot)$ in (1)–(2) is set to be a sigmoidal function in both the encoder and decoder

$$f(x) = \frac{1}{1+e^{-x}}. \tag{8}$$

Tied weights are used while training the AE. Thus, there remain three groups of parameters to learn: W, b_y, and b_z.

Before setting the updating rule for the weights, we need to properly set the "error" rule—a cost function—first. There are a lot of ways to define such a cost function for reconstruction. Since sigmoid activation is used, the derivative $f'(x)$ tends to asymptotically close to 0 as the output of the neuron draws near 0 or 1. If we use an ordinary cost like mean-square error, the gradient of the cost will also suffer from the same problem, which results in an unacceptably slow training speed. However, it can be found from the following derivation that cross-entropy tends to allow errors to change weights even when nodes saturate (i.e., outputs are close to 0 or 1.). So we always use cross-entropy when activation is set to be sigmoidal.

In our implementation, the cost is actually computed on a mini-batch of inputs since we adopt a mini-batch update strategy for the large dataset

$$c = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{d}[x_{ik}log(z_{ik}) + (1-x_{ik})log(1-z_{ik})]. \tag{9}$$

where d denotes the input vector size and m denotes the mini-batch size. $x_{ik}(z_{ik})$ denotes kth element of the ith input (reconstruction) in the mini-batch. The inner summation is over the input dimension, whereas the outer is over a whole mini-batch.

We optimize (9) using the mini-batch stochastic gradient descent method. We will now derive the partial differentials of cost with respect to parameters W, b_y, and b_z. First, we will rewrite the reconstruction in a scalar form

$$net_{ip}^{y} = \sum_{q=1}^{d} x_{iq}W_{qp} + b_{yp} \tag{10}$$

$$net_{ik}^{z} = \sum_{p=1}^{h} W_{kp}f(net_{ip}^{y}) + b_{zk} \tag{11}$$

$$z_{ik} = f(net_{ik}^{z}) = f\left(\sum_{p=1}^{h} W_{kp}f\left(\sum_{q=1}^{d} x_{iq}W_{qp} + b_{yp}\right) + b_{zk}\right) \tag{12}$$

where net_{ip}^{y} (net_{ik}^{z}) denotes the net input of the pth hidden (output) unit, given the ith sample in the mini-batch.

It is explicit that the first-order and second-order derivative of (8) are

$$f'(x) = f(x)[1-f(x)] \tag{13}$$

$$f''(x) = f(x)[1-f(x)][1-2f(x)]. \tag{14}$$

Having deducted (10)–(12), we can compute the partial derivatives of the reconstruction z over parameters W, b_y, and b_z. To simplify notations, we still show these equations in a scalar form

$$\frac{\partial z_{ik}}{\partial W_{rs}} = \begin{cases} f'(net_{ik}^{z})[W_{ks}f'(net_{is}^{y})x_{ir} + f(net_{is}^{y})] & (k=r) \\ f'(net_{ik}^{z})[W_{ks}f'(net_{is}^{y})x_{ir}] & (k\neq r) \end{cases} \tag{15}$$

$$\frac{\partial z_{ik}}{\partial b_{yr}} = f'(net_{ik}^{z})W_{kr}f'(net_{ir}^{y}) \tag{16}$$

$$\frac{\partial z_{ik}}{\partial b_{zr}} = f'(net_{ik}^{z}) \tag{17}$$

where W_{rs} means the weight connecting the rth input and the sth hidden unit. b_{yr} (b_{zr}) stands for bias of the rth unit in the hidden (reconstruction) layer.

Putting them (10)–(17) all together, we have partial differentials of cost (9) over parameters W, b_y and b_z

$$\begin{cases} \frac{\partial c}{\partial W_{rs}} = -\frac{1}{m}\sum_{i=1}^{m}\left\{\sum_{k=1}^{d}\left[\frac{x_{ik}-z_{ik}}{z_{ik}(1-z_{ik})}f'(net_{ik}^{z})W_{ks}f'(net_{is}^{y})x_{ir}\right] \right. \\ \qquad \left. +f'(net_{ik}^{z})f(net_{is}^{y})\right\} \\ \frac{\partial c}{\partial b_{yr}} = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{d}\frac{x_{ik}-z_{ik}}{z_{ik}(1-z_{ik})}f'(net_{ik}^{z})W_{kr}f'(net_{ir}^{y}) \\ \frac{\partial c}{\partial b_{zr}} = \frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{d}\frac{x_{ik}-z_{ik}}{z_{ik}(1-z_{ik})}f'(net_{ik}^{z}). \end{cases} \tag{18}$$

Substituting them into (5)–(7), the weight updating rules are determined.

After training the network, we remove the reconstruction layer and deem the hidden activity to be the learned feature. Subsequent layers are trained in the same manner, but their inputs are instead the outputs of their former layers. The SAE is thus constructed with encoders layer by layer.

B. Fine-Tuning and Classification

To integrate the layers of neural networks and perform classification by utilizing the learned feature, we need to fine-tune the whole pretrained network with a logistic regression classifier, which uses soft-max as its output-layer activation. Soft-max ensures the activation of each output unit sums to 1, so that we can deem the output as a set of conditional probabilities. For example, given input vector R, which is an output of former layers of AEs, the probability that the input belongs to category i equals

$$P(Y=i|R,W,b) = s(WR+b) = \frac{e^{W_iR+b_i}}{\sum_j e^{W_jR+b_j}} \tag{19}$$

where W and b are weights and biases of the logistic regression layer, and the summation is over all the output units.

The output-layer size is set to be the same as the total number of classes, and the input has the same size as the dimension of last-layer features. Since the logistic regression is implemented as a single-layer neural network, it can be merged with the former layers of networks to get a deep classifier. The fitting of the classifier is conducted over the whole architecture, but with very

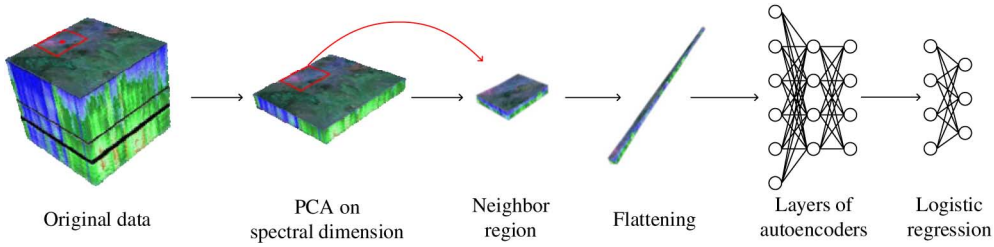


Fig. 4. Spatial-dominated information classification scheme. The first step of procesing is PCA compressing over spectral dimension, then after flatening the data, AEs are introduced to extract layer-wise deep features.

slight learning rates on former layer AEs. Expressions for the partial derivatives can be very complicated, but deducing them is similar to those in Section III-A.

In a nutshell, the whole flowchart of the proposed SAE-LR algorithm is as follows.

Algorithm 1: SAE With Logistic Regression (SAE-LR)

begin
  initialize mini-batch size b, pretraining epochs **pt**, pretraining learning rate **pl**, fine-tuning epochs **ft**, fine-tuning learning rate **fl**, number of layers d and number of neurons in each hidden layer **n[d]**. Input dimension D, total number of classes C.
  for every layer L ($1 \leq L \leq d$)
    Construct an autoencoder with **d_vis** input neurons, **d_hid** hidden neurons.
    if L is the first layer (i.e., $L = 1$)
      $d\_vis = D$
      $d\_hid = n[1]$
      Set input of the autoencoder x to be initial data.
    else
      $d\_vis = n[L-1]$
      $d\_hid = n[L]$
      Set input of the AE x to be the output of its former layer.
    end
    initialize AE weight matrix W with random variables, and biases b_y and b_z as zeros.
    for every pretraining epoch
      for every mini-batch
        Compute reconstruction:
        $$z = f(Wf(Wx + b_y) + b_z)$$
        Compute cost:
        $$c = -\frac{1}{b}[xlog(z) - (1-x)log(1-z)]$$
        Update weights using (18), with learning rate **pl**.
      end
    end
    Cast away the reconstruction layer.
  end
  initialize logistic regression layer input neurons as **n[d]**, output neurons as C.
  for every fine-tuning epoch
    for every mini-batch
      Compute probability of each class according to (19).
      Update weights from top layer to the bottom using ordinary back propagation, with learning rate **fl**.
    end
  end
end

IV. Classifying With Spatial-Dominated Features

Unlike other hyperspectral data spatial information extraction methods which only use the four or eight tangent neighbors or simple filtering, our deep framework takes all the pixels in a flat neighbor region into consideration, and lets the AEs learn the feature by itself. The overall flowchart of our proposed method is detailed in Fig. 4.

We propose to take all voxels in a neighborhood region of a certain pixel in the original data into consideration. Due to the hundreds of channels along the spectral dimension, it always has tens of thousands of dimensions. A large neighborhood region will result in too large input dimension for the classifier, containing too large amount of redundancy. So, in the first layer, PCA is introduced to condense the whole image, to reduce the data dimension to an acceptable scale and in the meantime reserving spatial information. Since we mainly care about incorporating spatial information in this method, we use PCA along the spectral dimension and only retain the first several principle components. The PCA transformation matrix is fitted on the whole image, both for tagged and untagged pixels. This step does cast away part of the spectral information, but since PCA is conducted for pixel vectors, the spatial information remains intact. Then, in the second layer, we extract a neighborhood region of the pixel in this condensed

data, which has only several principle components in its spectral dimension. This layer yields a total dimension of several hundreds and that is acceptable.

After these processes, we "flatten" the data in the third layer, i.e., stretch it to a 1-D vector, and feed it into a SAE. Subsequent layers include a layer-wise training SAE and fine-tune the whole model with logistic regression. These steps are similar to the former subsection which deals with deep spectral feature, and thus we will not repeat describing them here.

The procedures that our algorithm consists of are detailed below.

Algorithm 2: Classification With Spatial-Dominated Feature

1. **begin**
2. **initialize** neighborhood region size $\boldsymbol{a}$, number of principle components $\boldsymbol{n}$, image height $\boldsymbol{h}$, width $\boldsymbol{w}$.
3. PCA transform the image.
4. Retain the first $\boldsymbol{n}$ principle components. Thus we have an image of $\boldsymbol{h} \times \boldsymbol{w} \times \boldsymbol{n}$ size.
5. **for** each pixel
6. Crop a neighboring region for each point.
7. For those points near the edge that don't have enough surrounding pixels, fill in with its mirror.
8. Flatten the $\boldsymbol{a} \times \boldsymbol{a} \times \boldsymbol{n}$ array into a vector of $\boldsymbol{a}^2\boldsymbol{n} \times \mathbf{1}$ size.
9. **end**
10. Concatenate all vectors to form a matrix $\mathbf{M}$.
11. Train a SAE-LR with $\mathbf{M}$ as the input. Training procedures are the same to Algorithm 1.
12. **end**

V. Joint Spectral–Spatial Classification Framework

In this section, we integrate the spectral and spatial-dominated features together to construct a joint spectral–spatial classification framework. The whole flowchart is shown in Fig. 5.

The spectrum of a pixel should first be taken into consideration, since it contains the most important information for discriminating different kinds of ground categories. For spatial information, we extract the first several principle components of a neighborhood region to get the spatial-dominated information, which helps improve classification accuracy as verified in Section IV. This procedure corresponds to the first three steps of processing in Section IV (Fig. 4). These coefficients are then concatenated to the spectrum of that pixel, forming a hybrid set of features consisting of both spectral and spatial information.

Following training and fine-tuning steps mentioned above, we can eventually get class labels for each pixel. The whole flowchart of this final framework is shown in the following chart.

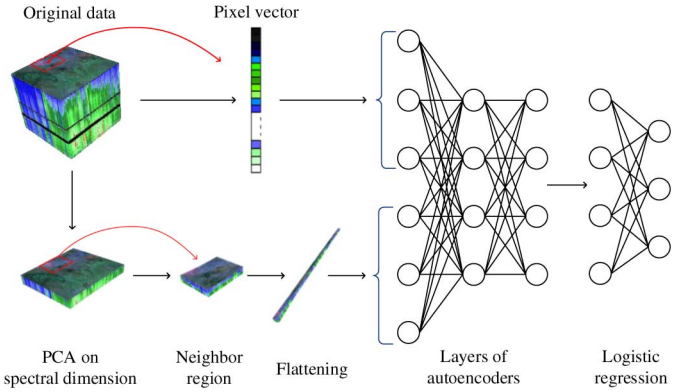


Fig. 5. Joint spectral–spatial classification framework. Spectral and spatial information are extracted separately via the former mentioned schemes, and feature extraction is conducted via a deep architecture like SAEs. Final classification is implemented as the final layer of the neural network, using classical neural network classifiers like logistic regression.

Algorithm 3: Joint Spectral–Spatial Classification

1. **begin**
2. Extract Spatial-dominated feature for each pixel according to Algorithm 2 to form a matrix $\mathbf{M}$.
3. Scale $\mathbf{M}$ into unit interval.
4. Normalize the whole initial image onto unit interval.
5. **for** each pixel
6. Add spectrum of each pixel on tail of each pixel's feature vector, (i.e., rows in $\mathbf{M}$).
7. **end**
8. Train a SAE-LR with $\mathbf{M}$ as the input. Training procedures are the same to Algorithm 1.
9. **end**

VI. Experimental Results

A. Data Description and Experiment Design

In our study, two hyperspectral datasets with different environmental settings are used to validate our proposed methods. They are the mixed vegetation site over Kennedy Space Center (KSC), FL, USA, and an urban site over the city of Pavia, Italy.

The first study site lies around KSC, FL, USA (Fig. 6). The image data was acquired by the National Aeronautics and Space Administration (NASA) Airborne Visible/Infrared Imaging Spectrometer instrument, on March 23, 1996. AVIRIS acquires data in a range of 224 bands with wavelengths ranging from 0.4 to 2.5 μm. The KSC data with 512×614 pixels has a spatial resolution of 18 m. Given water absorption and low signal-to-noise ratio (SNR) bands, 176 spectral bands are used for classification with 48 bands discarded. 13 different land-cover classes available in the original dataset are displayed in Table I.

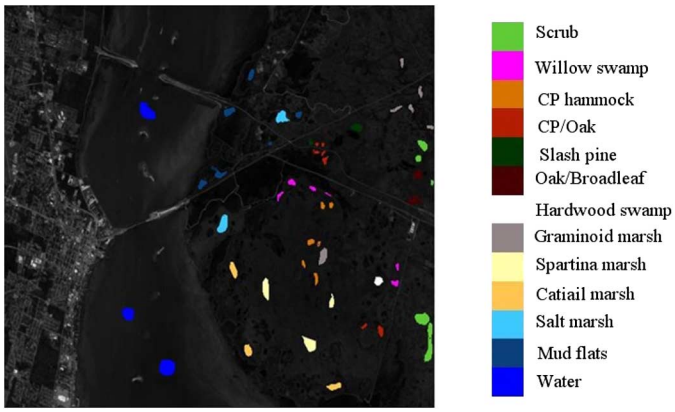


Fig. 6. NASA data, KSC. Band 20 and corresponding ground truth areas representing 13 land cover classes.

TABLE I
LAND COVER CLASSES AND NUMBERS OF PIXELS IN KSC DATASET

Class code	Name	No. of training samples	No. of validation samples	No. of testing samples
1	Scrub	457	152	152
2	Willow swamp	140	52	51
3	CP hammock	141	62	53
4	CP/Oak	159	51	42
5	Slash pine	95	30	36
6	Oak/Broadleaf	147	46	36
7	Hardwood swamp	56	25	24
8	Graminoid marsh	259	86	86
9	Spartina marsh	283	128	109
10	Cattail marsh	243	87	74
11	Salt marsh	249	74	96
12	Mud flats	300	102	101
13	Water	556	181	190
	Total	3100	1100	1050

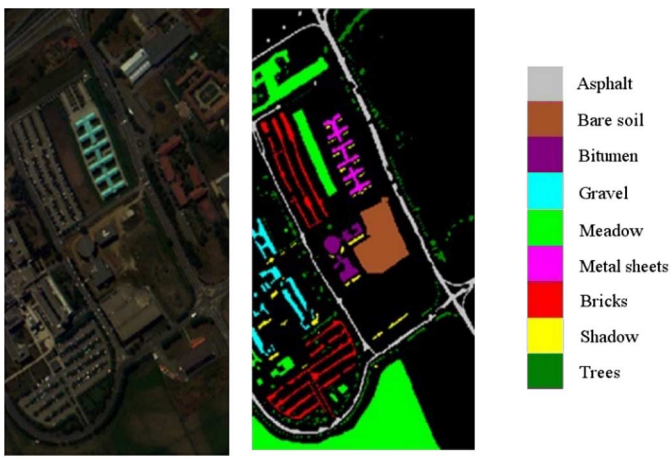


Fig. 7. ROSIS-3 data, Pavia, Italy. False-color composite (Band 5, 28, 56) and representing nine land cover classes.

TABLE II
LAND COVER CLASSES AND NUMBERS OF PIXELS IN PAVIA DATASET

Class code	Name	No. of training samples	No. of validation samples	No. of testing samples
1	Asphalt	4110	1370	1371
2	Bare soil	11 212	3735	3739
3	Bitumen	1324	441	442
4	Gravel	2062	689	685
5	Meadows	854	268	256
6	Metal sheets	3066	1016	1022
7	Bricks	824	274	258
8	Shadow	2345	781	752
9	Trees	626	192	208
	Total	25 550	8550	8450

The second dataset is gathered by a sensor known as the reflective optics system imaging spectrometer (ROSIS-3) over the city of Pavia, Italy, with 610×340 pixels (Fig. 7). 115 bands are collected in the $0.43-0.86$ μm range of the electromagnetic spectrum. The high spatial resolution of 1.3 m per pixel aims to avoid a high fraction of mixed pixels. In the experiment, some bands have been removed due to noise; the remaining 103 channels are used for the classification. Nine land cover classes are selected, which are shown in Fig. 7 and the numbers of samples for each class are displayed in Table II.

In both images, we split the tagged parts of the image into three sets, i.e., training, validation, and testing data, with a split ratio 6:2:2. That is, we randomly choose 60% of the tagged samples as the training set, and 20% and 20% for the validation and testing sets, respectively. During training, we use the training set to learn weights and biases of each neuron and use the validation set to tune the best super-parameters like hidden unit sizes or hidden layer numbers. The test set is used to produce final classification results. Thus, small classes will be trained and tested with a smaller number of pixels in contrast with large classes.

Experiments were organized into four parts. The first aims at analyzing the behavior of AEs, which are the building blocks of our proposed methods. In the second experiment, the effectiveness of deep architecture is tested in comparison to the SVM-based method. In the third part, we test the classification accuracy of spatial-dominated features of hyperspectral data. Finally, the effectiveness of joint spatial–spectral feature, which is the best of all proposed models, is inspected.

In order to quantitatively compare and estimate the capabilities of the proposed models, overall accuracy (OA), average accuracy (AA), and Kappa coefficient [45] are used as performance measurement. In the experiments, we split the dataset into three parts, i.e., training, validation and testing data, and apply cross-validation analysis with the KSC and Pavia datasets [46].

To perform statistical evaluation, we conduct 100 independent replications of the whole process and use the average Kappa coefficient to compare the performance between different methods. As mentioned above, for each replication, the training, validation, and testing data are randomly selected with a ratio of 6:2:2. A paired t-test is performed to test whether the observed increase in the mean Kappa coefficient is statistically significant (at the level of 95%) [46].

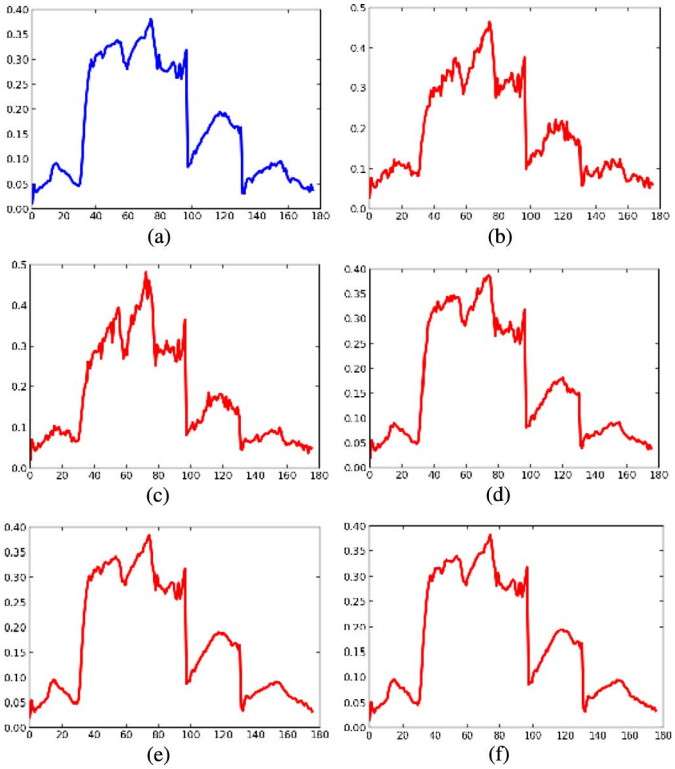


Fig. 8. Reconstructions of a same input in different iteration epochs. (a) Input spectrum. (b)–(f) Reconstructions of (a) in epoch 1, 10, 100, 1000, and 3500, respectively. Vertical axis stands for normalized reflectance, whereas horizontal axis stands for band numbers.

B. AEs: Behavior and Analysis

Single layer AEs are basic building blocks of our proposed models, so we investigate the behavior of AEs in this section before we present classification accuracies of more complicated models.

1) Reconstruction: First, we examine the quality of the extracted features by checking the quality of the reconstructed spectra. We use a single-layer AE with 100 hidden units and train it on the KSC dataset. It is shown that the AE do progressively learn better reconstruction during training. Since the AE restitutes a rather perfect reconstruction from hundreds of iterating epochs (Fig. 8) and computing the reconstruction need only the hidden activity (Section II-B), we can say that the learned hidden activity retains enough information from the input. Thus, it can be thought as a good feature set for the input data.

2) Filters Learned: Suppose the dataset to be processed has N spectral bands, we are using an AE with N input neurons and H hidden neurons. The input-to-hidden layer of an AE is fully connected, so every single hidden unit has its connections to every input neuron. For each hidden unit, it has a fan-in of N connections. The N connections as a whole can be viewed as a "filter" since they behave by filtering away information from some input which represent certain wavelengths and at the same time exaggerating others. In this way, an AE learning with H hidden units can be viewed as learning with H such filters.

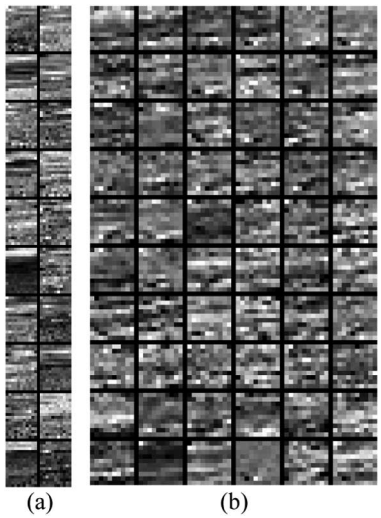


Fig. 9. Filter images learned by an AE on (a) KSC dataset and (b) Pavia dataset. Each N-pixel tiny rectangle stands for N input-to-hidden weights that connects each input unit to a same hidden unit. The intensity of each pixel stands for the absolute value of corresponding weights.

There is a convenient way of visualizing these filters. We truncate the weight vector into pieces of equal length, and vertically concatenate them to form a matrix M. So the matrix M has N entries and for the whole network, we have H such matrixes. Then, for each matrix M, we use the intensities of N pixels of a tiny image patch which has the same size as M (called "filter image") to reflect the N connection. By plotting a filter image for each hidden unit, we can observe some interesting features of these learned filters more conveniently (Fig. 9).

Fig. 9(a) and (b) shows the filters acquired after training AE s on KSC and Pavia datasets, respectively. Some hidden units have large weights over a small portion of input units and small weights over others, which suggest that a certain wavelength interval is informative and discriminative and others' weights have more complex connecting patterns, having ripples over different input units or showing Gaussian-like noises in some bands. To make the visualization more direct, these 1-D connections are horizontally folded into 16×11 and 10×11 pixels corresponding with the 176 and 103 input sizes of the KSC and Pavia data. That is why we find all filters are extracting "horizontal" features in all of the plotted filter images. In Fig. 9(a), there are 20 hidden units in the trained AE, thus we can see 20 tiny filter images in the plot. For the Pavia data, the situations are similar [Fig. 9(b)], but with 60 hidden units.

3) Running Time: We concede that neural networks take longer time to train compared with other machine learning algorithms like KNN or SVM, and so does deep learning. In this section, we focus on how the running time changes with respect to the scale of AE model.

First, we inspect the training time. We use 3100 training samples for each AE on a NVIDIA GT750M graphics card. The pretraining epochs are set to be 5000, whereas fine-tuning epochs are set to be 50 000. Experimental results [Fig. 10(a)] show that training time generally grows with the increase of input and hidden sizes. On the contrary, if we keep fix hidden size or input size, training time grows proportionally with respect to training epochs.

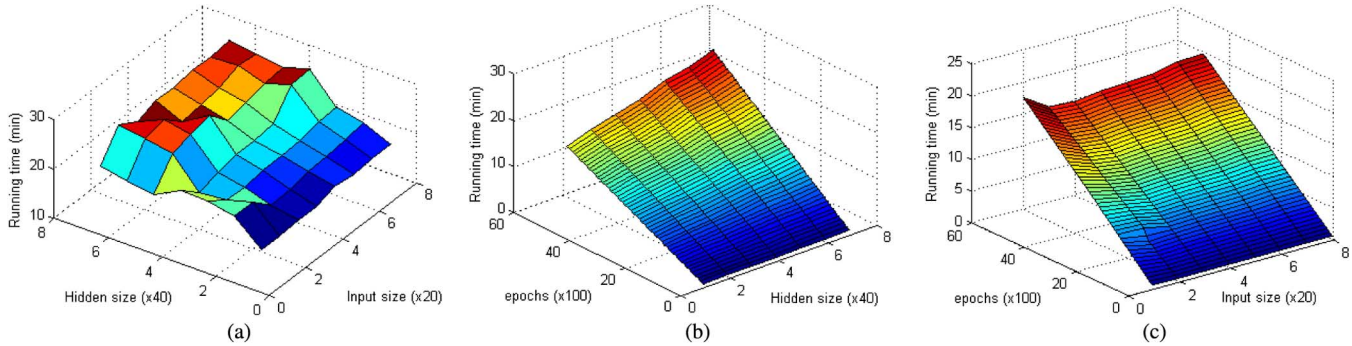


Fig. 10. Factors influencing training time. (a) Training time of an AE with different hidden and input sizes. (b) Training time elapsed on each epoch whereas varying hidden sizes. (c) Training time elapsed on each epoch whereas varying input size.

TABLE III
TESTING TIME COMPARISON

Classifiers	Testing time (s)	
	KSC	Pavia
AE-LR	1.14	1.18
RBF-SVM	112.19	175.05
Linear SVM	52.32	232.97
PCA RBF-SVM	12.29	189.60
KNN	35.71	222.28

On the other hand, an advantage of deep learning algorithms is that they are super-fast on testing. In Table III, an AE of hidden size 20 on the KSC dataset and 60 on the Pavia dataset with logistic regression is compared with radial basis function (RBF), kernel SVM, linear SVM, and k nearest neighbors. We take all 314 368 pixels in the KSC dataset and 207 400 pixels in the Pavia dataset for classification and compare the running time of all the mentioned classifiers. Experiments in both the two dataset have confirmed that AE runs much faster than other classification algorithms in the control group.

4) Comparing With Other Feature Extraction Methods: By comparing the AEs with other feature extraction methods, involving principle component analysis (PCA), kernel PCA (KPCA), independent component analysis (ICA), nonnegative matrix factorization (NMF), and factor analysis (FA), we verify the effectiveness of these AE features from the sense of classification.

First, we substitute the AE in the SAE-LR scheme with these feature extraction methods. All the logistic regression classifiers are set to have learning rate 0.1 and are iterated on the training data for 10 000 epochs. The SAE only consists of one layer of AE. Experiments show that by combining with logistic regression, AE outperforms all other feature extraction methods and gets the highest accuracy.

To be fair, we also combine the aforementioned feature extraction methods with SVM to verify if AEs bring more benefits for classification. Results show that although logistic regression as a neural network tends to be more sensitive to dimensions, AEs help improve the accuracies of both classifiers. The only exception lies in the AE-SVM case, where factor analysis outperforms AE with 20 extracted features [Fig. 11(b)].

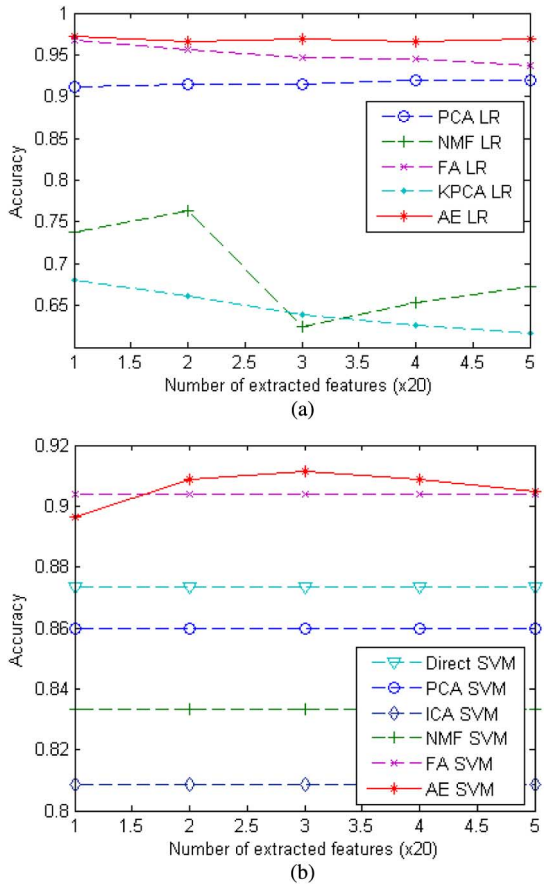


Fig. 11. (a) AE-LR and (b) AE-SVM performance with respect to hidden sizes on the KSC dataset. Dashed lines stands for performance of the control group and the red solid line stands for AE-based methods. Horizontal axis stands for the number of features we extract in the control group and number of hidden units we use while training an AE. In ICA, we choose the parallel fast ICA algorithm and use initial whitening as the preprocessing step, and the maximum iteration step is set to be 200. In NMF, we use the projected gradient method and we use RBF kernel in KPCA.

C. Classification With Spectral Feature

In Section VI-B, we have examined various characteristics of AEs. In this part of experiment, we begin to exploit their potential by applying them purely to spectral information. Here, we mainly focus on the effect of depths in order to compare with the typical classifier SVM.

TABLE IV
IMPACT OF DEPTH

Depth	KSC		Pavia	
	Overall test set accuracy (%)	Running time on test set (s)	Overall test set accuracy (%)	Running time on test set (s)
1	94.63	0.12	92.93	0.19
2	95.45	0.15	94.95	0.27
3	**96.55**	0.20	94.99	0.35
4	95.27	0.22	**95.16**	0.42
5	93.91	0.24	95.13	0.48

TABLE V
CLASSIFYING WITH SPECTRAL FEATURES

Datasets	Measurements	SAE-LR	Linear SVM	PCA RBF-SVM	RBF-SVM
KSC	OA	**0.9673**	0.9552	0.9535	0.9651
	AA	**0.9408**	0.9197	0.9157	0.9395
	Kappa	**0.9636**	0.9501	0.9482	0.9611
Pavia	OA	**0.9514**	0.9111	0.9448	0.9460
	AA	**0.9401**	0.8758	0.9274	0.9340
	Kappa	**0.9370**	0.8835	0.9282	0.9299

1) Effect of Depth: Depth plays an important role in the classification accuracy because it determines the quality of feature from various aspects like invariance and abstraction. In Table IV, we tried several SAEs with different depths. For the KSC data, it has 176 spectral channels and each hidden layer size is set to 20, and also a logistic regression layer is used on top of the SAE. So the neural networks are constructed as 176-20-$\cdots$-20 $-$ 13. For the Pavia dataset which has 103 spectral channels and 9 classes, the performance reaches its best when using 60 as the size of hidden layers. The neural networks are like 103-60-$\cdots$-60-9. "Depth" corresponds to the number of 20 or 60-sized layers in the deep neural network. Experiments show that depth does help to increase classification accuracy. Note that in this part of experiment, the AE is not fully tuned, with only 5000 epochs of pretraining and 50 000 epochs of fine-tuning. If we continue training the model, it will yield higher accuracies.

2) Comparison With SVMs: For comparison with the classical SVM models, we conduct SVM with linear kernel, RBF-SVM with PCA feature extraction, and SVM with RBF kernel on the KSC and Pavia data. The SVM parameters are tuned to achieve the best performance, as is the SAE-LR classifier. To elaborate, the SAE-LR trained for KSC data has 20 hidden units and 1 hidden layer and is trained with 3300 epochs of pretraining and 400 000 epochs of fine-tuning with a very slight learning rate. SAE-LR for the Pavia dataset is constructed similarly, but with 4 layers and 60 hidden units per layer.

We performed the experiments with same parameter settings as described above 100. Table V shows the mean value of the OA, AA, and Kappa coefficients for the 100 replications. We can see that out of all the methods in the control group, RBF-SVM yields the highest accuracy in terms of mean OA. However, the SAE-LR method turns out to be better than RBF-SVM on all three measurements. It has shown a significant gain in accuracy

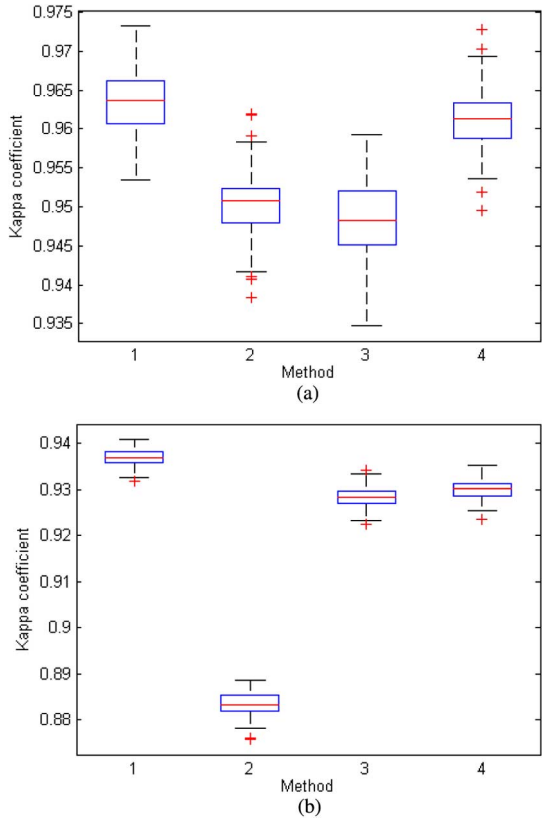


Fig. 12. Box plot of Kappa coefficients of different methods on (a) KSC and (b) Pavia datasets. Numbers in the abscissa corresponding to 1) SAE-LR; 2) Linear SVM; 3) PCA RBF-SVM; and 4) RBF-SVM. We plot these boxes by doing 100 independent replications. The red line through the center of each box indicates the median value of the Kappa coefficients. The edges of boxes are the 25th and 75th percentiles. Whiskers extend to the maximum and minimum points. Abnormal outliers shown as red " +"s.

for both the KSC and Pavia datasets. Further, we performed the paired t-test (as described in Section VI-A) between the trained SAE-LR and the other three SVM models. The detailed statistics of the Kappa coefficients of the four methods are shown in Fig. 12. Paired t-test results show that improvements on Kappa coefficients are statistically significant (at the level of 95%).

In the following sections VI-D and VI-E of experiments, we will show that if spatial information is incorporated, classification results will grow much higher, and thus further exceed RBF-SVM's accuracy.

D. Classification With Spatial-Dominated Feature

If we directly apply SVM on the spatial-dominated information collected by cropping adjacent patches, the accuracy will be slightly higher than that yielded by spectral features (Table VI). What is more, our deep neural networks confirm that these kinds of features can lead to higher accuracy for classification in terms of mean performance. We inspect our spatial information extraction method by varying the number of retained principle components and depth of neural network.

1) Differing Principle Components: Although the proposed spatial-dominated method majorly focuses on extracting spatial information of hyperspectral data, using how much spectral information to retain still plays a role in the completeness of

TABLE VI
SPATIAL-DOMINATED AND JOINT CLASSIFICATION OF SAE-LR AND SVM MODEL

Datasets	Measurements	SAE-LR		RBF-SVM		EMP RBF-SVM
		Spatial-dominated	Joint	Spatial-dominated	Joint	
KSC	Overall accuracy	0.9776	**0.9876**	0.9708	0.9867	0.9853
	Average accuracy	0.9625	**0.9790**	0.9538	0.9769	0.9704
	Kappa coefficient	0.9750	**0.9862**	0.9675	0.9852	0.9836
Pavia	Overall accuracy	0.9812	**0.9852**	0.9725	0.9741	0.9791
	Average accuracy	0.9732	0.9782	0.9666	0.9620	**0.9798**
	Kappa coefficient	0.9755	**0.9807**	0.9642	0.9663	0.9727

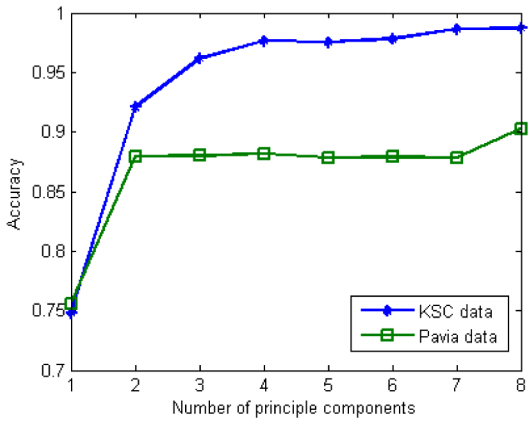


Fig. 13. Effect of principle components.

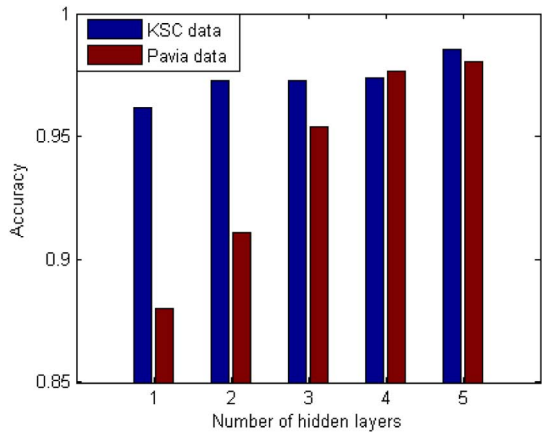


Fig. 14. Effect of depth.

its features. The amount of spectral information can be adjusted by varying the number of principle components (i.e., the n in Algorithm 2). Here, we vary the retained principle components from 1 to 8, and check how the final classification accuracy is affected. In Fig. 13, an SAE-LR model with one hidden layer is constructed. It shows that as the number of principle components grows, the classification accuracies of both images become higher. As a trade-off between accuracy and data size, we empirically choose 4 as the number.

2) Effect of Depth: Depth also plays an important role in spatial-dominated classification. We train a series of SAEs with different depths, but with fixed principle component numbers and hidden unit numbers to see how the depth of the features effect classification accuracies. Results are show in Fig. 14. Compared with spectral information, deeper features are required for spatial-dominated information to get the best classification accuracy. This helps us to determine how many layers are needed to get an optimal configuration of the network.

E. Joint Spectral–Spatial Classification Framework

This section of experimentation culminates all of our previous methods. By putting both the spectral and spatial information together to form a hybrid input and utilizing the deep classification framework detailed in Section IV-B, we get the highest classification accuracy we have ever attained.

1) Comparison With Spatial-Dominated Methods and SVMs: Here, we compare joint spectral–spatial classification with the aforementioned spatial methods. We also perform RBF-SVM on both to form a control group (Table VI). Similarly, experiments are also performed for 100 times. Compared with Table V, we can figure that for both the SAE-LR and RBF-SVM methods, joint features yield higher accuracy than spectral features in terms of mean performance, and while comparing the two methods within each feature set, SAE-LR is more precise. As in the last Section VI-E2, statistical evaluations of Kappa coefficients are plotted as the left four boxes in Fig. 15(a) and (b). We also performed paired t-tests between SAE-LRs and their corresponding control group SVMs. The results have shown that SAE-LR does achieve higher accuracy.

2) Comparing With Other Spatial Methods: Spatial information is very important in hyperspectral data classification. Some methods such as extended morphological profile (EMP) try to integrate spatial information into spectral-based classifiers. In the EMP method, principle components of hyperspectral data are computed and then the morphological profiles are used to extract spatial information on the first several components. EMP followed by SVM is a successful spatial–spectral classification method of hyperspectral data. We searched a range of c and g configurations for the SVM used in the EMP RBF-SVM method, and for the KSC data, they are configured as $c = 100$ and $g = 1$, whereas those in the Pavia data are $c = 100$, $g = 10$ and performed 100 replications [the rightmost column in Fig. 16(a) and (b)]. Paired t-tests also show that the joint information-based SAE-LR does consistently reach a higher accuracy than the EMP RBF-SVM.

3) Whole Image Classification: In this section, we examine the classification accuracy from a visual perspective. We choose the best SAE-LR models for the spectral, spatial-dominated, and joint sets of information to classify the whole images of KSC and Pavia. All parameters in these models are optimized. From

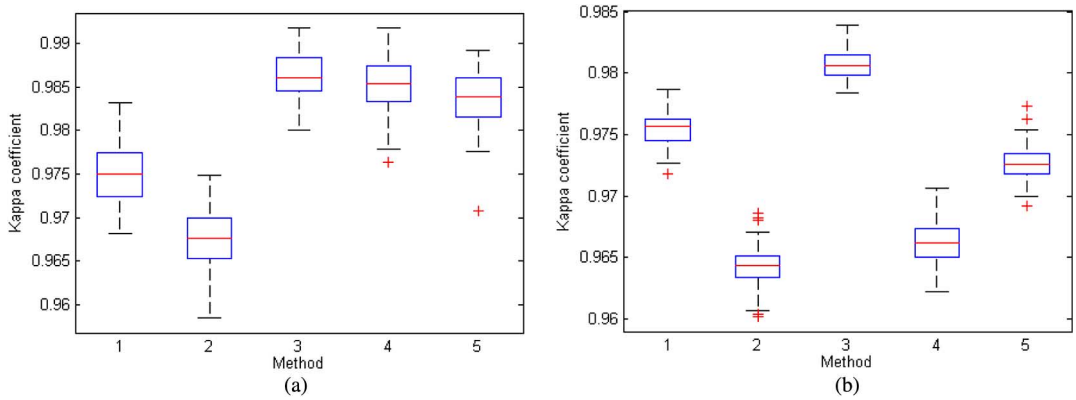


Fig. 15. Box plot of Kappa coefficients of spectral, spatial-dominated and joint classification scheme on (a) KSC and (b) Pavia datasets. Numbers in the abscissa corresponding to 1) SAE-LR on spatial-dominated information; 2) RBF-SVM on spatial-dominated information; 3) SAE-LR on joint information; 4) RBF-SVM on joint information; and 5) EMP RBF-SVM. The meanings of the indicators in the box are the same as Fig. 12.

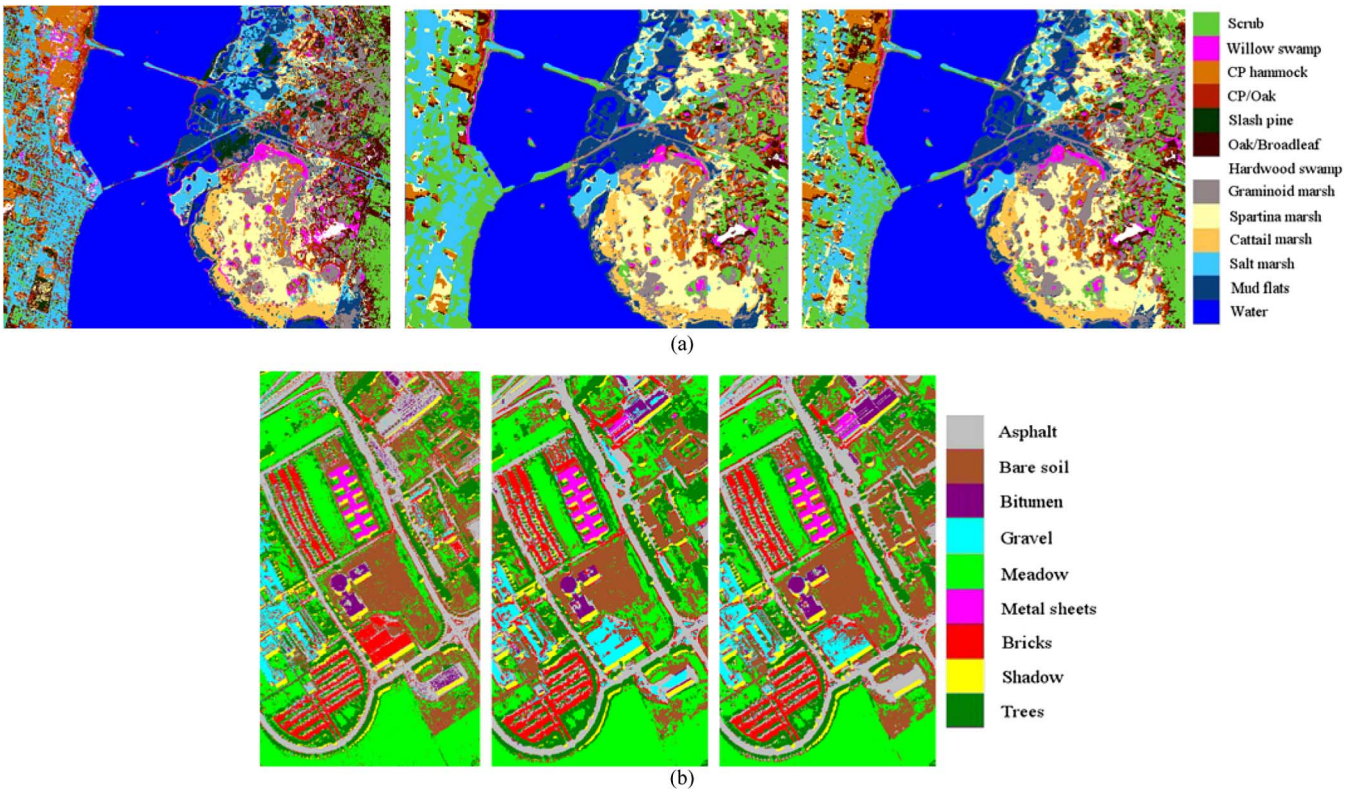


Fig. 16. Spectral (left), spatial-dominated (middle), and joint (right) classification results of the whole image on (a) KSC and (b) Pavia datasets. Results are generated with learned SAE-LR models.

the resulting images, we can figure out how the proposed spatial-dominated information extraction method affects the classification results. In Fig. 16, from both images, we can see that spectral classification always results in noisy scatter points in the image (left) and that spatial-dominated features correct this shortcoming (middle). However, spatial-dominated features have their own flaws. They misclassify certain small regions like the shadow of the third line of buildings in the Pavia data. Because of its window size, some details of targets are lost. This can be found in the area of bare soil on the lower-right side of the Pavia data. Results yielded by spatial-dominated features totally lost the shape of the bare soil region, whereas for spectral results, the shape is retained. Finally, for the joint classification, it gives a satisfying trade-off. It retains the shape and detail of some objects, while simultaneously eliminating noisy scattered points of misclassification.

VII. Discussion and Conclusion

In this paper, we propose hyperspectral data classification methods using deep features extracted by SAEs. It is shown that AE-extracted features are useful for classification, and it helps to

increase the accuracy of SVM and logistic regression while obtaining the highest accuracy when compared with other feature extraction methods like PCA, KPCA, and NMF.

For hyperspectral data classification, our proposed SAE-LR method has been proven to provide statistically higher accuracy than RBF-SVM, a classical classifier previously considered to be state-of-the-art in this field. In addition, we also inspected the impact that the depth of feature has on classifying hyperspectral data. Our experimental results suggest that deeper features always lead to higher classification accuracies, though too deep structure will act inversely. Based on our results, we suggest using 4–6 hidden layers of AEs with 20–60 hidden units per layer for hyperspectral data classification tasks. The disadvantage of SAE-LR is its training time, but in compensate, the testing time efficiency is much faster than other methods like SVM or KNN.

For our proposed spatial-dominated information-based classification, both SAE-LR and SVM have proved the effectiveness of the PCA-window spatial information extraction method. The SAE-LR classifier succeeds in classifying datasets and yields a higher accuracy than traditional spectral information-based methods.

Acknowledgment

The authors would like to thank Prof. R. Memisevic and Y. Bengio for their suggestion of using Theano, which accelerates the implementation of the models significantly. They would also like to thank the Editor who handled our paper and the three anonymous reviewers for providing truly outstanding comments and suggestions that significantly helped us improve the technical quality and presentation of our paper.

References

[1] D. Landgrebe, "Hyperspectral image data analysis," *IEEE Signal Process. Mag.*, vol. 19, no. 1, pp. 17–28, Jan. 2002.

[2] J. A. Richards, *Remote Sensing Digital Image Analysis: An Introduction*. New York, NY, USA: Springer, 2013.

[3] F. M. Lacar, M. M. Lewis, and I. T. Grierson, "Use of hyperspectral imagery for mapping grape varieties in the Barossa Valley, South Australia," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Sydney, Australia, 2001, vol. 6, pp. 2875–2877.

[4] F. V. D. Meer, "Analysis of spectral absorption features in hyperspectral imagery," *Int. J. Appl. Earth Observ. Geoinf.*, vol. 5, no. 1, pp. 55–68, Jan. 2004.

[5] P. W. Yuen and M. Richardson, "An introduction to hyperspectral imaging and its application for security, surveillance and target acquisition," *Imaging Sci. J.*, vol. 58, no. 5, pp. 241–253, May 2010.

[6] R. F. Egerton, *Electron Energy-Loss Spectroscopy in the Electron Microscope*. New York, NY, USA: Plenum, 1996.

[7] E. K. Hege *et al.*, "Hyperspectral imaging for astronomy and space surveillance," in *Proc. SPIE's 48th Annu. Meet. Opt. Sci. Technol.*, San Diego, CA, USA, 2004, pp. 380–391.

[8] A. A. Gowen *et al.*, "Hyperspectral imaging—An emerging process analytical tool for food quality and safety control," *Trends Food Sci. Technol.*, vol. 18, no. 12, pp. 590–598, Dec. 2007.

[9] T. J. Malthus and P. J. Mumby, "Remote sensing of the coastal zone: An overview and priorities for future research," *Int. J. Remote Sens.*, vol. 24, no. 13, pp. 2805–2815, Nov. 2003.

[10] J. Bioucas-Dias *et al.*, "Hyperspectral remote sensing data analysis and future challenges," *Geosci. Remote Sens. Mag.*, vol. 1, no. 2, pp. 6–36, Feb. 2013.

[11] S. Rajan, J. Ghosh, and M. M. Crawford, "An active learning approach to hyperspectral data classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 4, pp. 1231–1242, Apr. 2008.

[12] G. Camps-Valls and L. Bruzzone, "Kernel-based methods for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 6, pp. 1351–1362, Jun. 2005.

[13] G. M. Foody and A. Mathur, "A relative evaluation of multiclass image classification by support vector machines," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 6, pp. 1335–1343, Jun. 2004.

[14] A. Ambikapathi, T.-H. Chan, C.-H. Lin, and C.-Y. Chi, "Convex geometry based outlier-insensitive estimation of number of endmembers in hyperspectral images," in *Proc. IEEE Whispers*, Gainesville, FL, USA, Jun. 25–28, 2013.

[15] L. M. Bruce, C. H. Koger, and J. Li, "Dimensionality reduction of hyperspectral data using discrete wavelet transform feature extraction," *IEEE Trans. Geosci. Remote Sens.*, vol. 40, no. 10, pp. 2331–2338, Oct. 2002.

[16] L. O. Jimenez and D. A. Landgrebe, "Hyperspectral data analysis and supervised feature reduction via projection pursuit," *IEEE Trans. Geosci. Remote Sens.*, vol. 37, no. 6, pp. 2653–2667, Jun. 1999.

[17] J. C. Harsanyi and C. I. Chang, "Hyperspectral image classification and dimensionality reduction: An orthogonal subspace projection approach," *IEEE Trans. Geosci. Remote Sens.*, vol. 32, no. 4, pp. 779–785, Jul. 1994.

[18] C. I. Chang, Q. Du, T. Sun, and M. L. G. Althouse, "A joint band prioritization and band-decorrelation approach to band selection for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 37, no. 6, pp. 2631–2641, Jun. 1999.

[19] S. B. Serpico and L. Bruzzone, "A new search algorithm for feature selection in hyperspectral remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 39, no. 7, pp. 1360–1367, Jul. 2001.

[20] F. Samadzadegan, H. Hasani, and T. Schenk, "Simultaneous feature selection and SVM parameter determination in classification of hyperspectral imagery using Ant Colony Optimization," *Can. J. Remote Sens.*, vol. 38, no. 2. pp. 139–156, Mar. 2012.

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

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

[23] L. Zhuo *et al.*, "A genetic algorithm based wrapper feature selection method for classification of hyperspectral images using support vector machine," in *Proc. Geoinformat. Joint Conf. GIS Built Environ. Classif. Remote Sens. Images Int. Soc. Opt. Photonics*, Nov. 2008, pp. 71471J–71471J.

[24] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[25] N. Kruger *et al.*, "Deep hierarchies in primate visual cortex what can we learn for computer vision?" *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1847–1871, Aug. 2013.

[26] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Neural Inf. Process. Syst. 25*, Lake Tahoe, Nevada, USA, 2012, pp. 1106–1114.

[27] G. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313. no. 5786, pp. 504–507, Jul. 2006.

[28] D. Yu, L. Deng, and S. Wang, "Learning in the deep structured conditional random fields," in *Proc. Neural Inf. Process. Syst. Workshop*, Vancouver, BC, Canada, Dec. 2009, pp. 1–8.

[29] A. R. Mohamed, T. N. Sainath, and G. Dahl, "Deep belief networks using discriminative features for phone recognition," in *Proc. Acoust. Speech Signal Process. (ICASSP)*, Prague, Czech Republic, 2011, pp. 5060–5063.

[30] A. Plaza, J. Plaza, and G. Martin, "Incorporation of spatial constraints into spectral mixture analysis of remotely sensed hyperspectral data," in *Proc. IEEE Int. Workshop Mach. Learn. Signal Process.*, Grenoble, France, 2009, pp. 1–6.

[31] Y. Tarabalka, J. A. Benediktsson, and J. Chanussot, "Spectral-spatial classification of hyperspectral imagery based on partitional clustering techniques," *IEEE Trans. Geosci. Remote Sens.*, vol. 47, no. 8, pp. 2973–2987, Aug. 2009.

[32] M. Fauvel *et al.*, "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.

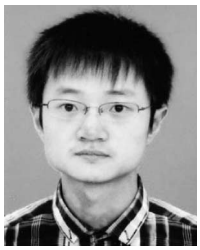
[33] J. Li, J. M. Bioucas-Dias, and A. Plaza, "Spectral-spatial classification of hyperspectral data using loopy belief propagation and active learning," *IEEE Trans. Geosci. Remote Sens.*, vol. 51, no. 2, pp. 844–856, Feb. 2013.

[34] J. Liu *et al.*, "Spatial-spectral kernel sparse representation for hyperspectral image classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 6, no. 6, pp. 2462–2471, Jun. 2013.

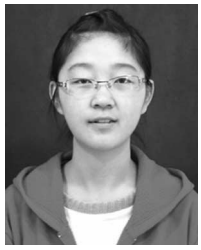
[35] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *Proc. Int. Joint Conf. IEEE Neural Netw.*, Washington, DC, USA, 1989, pp. 593–605.
[36] I. Sutskever and G. E. Hinton, "Deep, narrow sigmoid belief networks are universal approximators," *Neural Comput.*, vol. 20, no. 11, pp. 2629–2636, Nov. 2008.
[37] N. LeRoux and Y. Bengio, "Deep belief networks are compact universal approximators," *Neural Comput.*, vol. 22, no. 8, pp. 2192–2207, Aug. 2010.
[38] G. E. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006.
[39] R. Salakhutdinov and G. E. Hinton, "Deep Boltzmann machines," in *Proc. Int. Conf. Artif. Intell. Statist.*, Clearwater Beach, FL, USA, 2009, pp. 448–455.
[40] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Proc. Neural Inf. Process. Syst.*, Cambridge, MA, USA, 2007, pp. 153–160.
[41] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. Manzagol, "Stacked denoising autoencoders," *J. Mach. Learn. Res.*, vol. 11, no. 12, pp. 3371–3408, Dec. 2010.
[42] G. E. Hinton, "A practical guide to training restricted Boltzmann machines," Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, Tech. Rep. UTML TR2010-003, 2010.
[43] Y. LeCun *et al.*, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, Apr. 1989.
[44] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biol. Cybern.*, vol. 36, no. 4, pp. 193–202, Apr. 1980.
[45] W. D. Thompson and S. D. Walter, "A reappraisal of the kappa coefficient," *J. Clin. Epidemiol.*, vol. 41, no. 10, pp. 949–958, Oct. 1988.
[46] Z. Zhu, C. E. Woodcock, J. Rogan, and J. Kellndorfer, "Assessment of spectral, polarimetric, temporal, and spatial dimensions for urban and peri-urban land cover classification using Landsat and SAR data," *Remote Sens. Environ.*, vol. 117, pp. 72–82, Feb. 2012.
[47] Z. Zhen and G. Wang, "Learning discriminative hierarchical features for object recognition," *IEEE Signal Process. Lett.*, vol. 21, no. 9, pp. 1159–1163, Sep. 2014.

Yushi Chen received the Ph.D. degree from Harbin Institute of Technology, Harbin, China, in 2008.

Currently, he is an Assistant Professor with the School of Electrical and Information Engineering, Harbin Institute of Technology, China. He has published more than 20 peer-reviewed papers, and he is the inventor or coinventor of three patents. His research interests include hyperspectral data analysis, ensemble learning, deep learning and remote sensing applications.

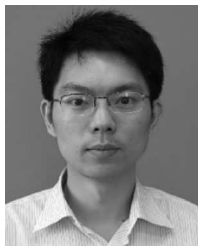


Zhouhan Lin received Bachelor's degree from Harbin Institute of Technology (HIT), Harbin, China, in 2012. Currently, he is pursuing the Master's degree at Institute of Image and Information Technology, HIT, in the field of parallel computing, machine learning, especially deep learning and their applications on remote sensing.



Xing Zhao received the Bachelor's degree from the College of Information and Communication Engineering, Harbin Engineering University, Harbin, China, in 2013. Currently, she is a Graduate Student at the Institute of Image and Information Technology, Harbin Institute of Technology.

Her research interests include hyperspectral image processing, machine learning, and deep learning.



Gang Wang received the B.S. degree in electrical engineering from Harbin Institute of Technology, Harbin, China, in 2005, and the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign (UIUC), Champaign, IL, USA, in 2010.

He is an Assistant Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore, and a Research Scientist with the Advanced Digital Science Center, Singapore. His research interests include computer vision, machine learning, object recognition, scene analysis, and large scale machine learning.

Dr. Wang is a recipient of the prestigious Harriett & Robert Perry Fellowship in 2009–2010 and CS/AI award in 2009 during the Ph.D. at UIUC.

Yangfeng Gu received the Ph. D degree in information and communication engineering from Harbin Institute of Technology, Harbin, China, in 2005.

He joined as a Lecture with the School of Electronics and Information Engineering, HIT. He was appointed as Associate Professor with the same institute in 2006. From 2011 to 2012, he was a Visiting Scholar with the Department of Electrical Engineering and Computer Science, University of California, Berkeley, CA, USA. Currently, he is a Professor with the Department of Information Engineering, Harbin Institute of Technology, Harbin, China. He has published more than 60 peer-reviewed papers, four book chapters, and he is the inventor or coinventor of seven patents. His research interests include image processing in remote sensing, machine learning and pattern analysis, and multiscale geometric analysis.

Dr. Gu was enrolled in first Outstanding Young Teacher Training Program of HIT. He is a peer Reviewer for several international journals such as IEEE TRANSACTION ON GEOSCIENCE AND REMOTE SENSING, IEEE TRANSACTION ON INSTRUMENT AND MEASUREMENT, IEEE GEOSCIENCE AND REMOTE SENSING LETTERS, and IET ELECTRONICS LETTER.