Title: Two-stream Deep Architecture for Hyperspectral Image Classification

This paper appears in: IEEE Transactions on Geoscience and Remote Sensing

Date of Publication: 2017

Author(s): Siyuan Hao; Wei Wang; Yuanxin Ye; Tingyuan Nie; Lorenzo Bruzzone

Volume:

Page(s):

DOI: 10.1109/TGRS.2017.2778343

Two-stream Deep Architecture for Hyperspectral Image Classification

Siyuan Hao, Wei Wang, Yuanxin Ye, *Member, IEEE*, Tingyuan Nie, Lorenzo Bruzzone, *Fellow, IEEE*

***Abstract*—Most traditional approaches classify hyperspectral image pixels only relying on the spectral values of the input channels. However, the spatial context around a pixel is also very important and can enhance the classification performance. In order to effectively exploit and fuse both the spatial context and spectral structure, we propose a novel two-stream deep architecture for hyperspectral image classification. The proposed method consists of a two-stream architecture and a novel fusion scheme. In the two-stream architecture, one stream employs the stacked denoising auto-encoder (SdAE) to encode the spectral values of each input pixel, and the other stream takes as input the corresponding image patch and deep convolutional neural networks (CNNs) are employed to process the image patch. In the fusion scheme, the prediction probabilities from two streams are fused by adaptive class-specific weights, which can be obtained by a fully-connected layer. Finally, a weight regularizer is added to the loss function to alleviate the overfitting of the class-specific fusion weights. Experimental results on real hyperspectral images demonstrate that the proposed two-stream deep architecture can achieve competitive performance compared with the state-of-art methods.**

***Index Terms*—Hyperspectral image classification, Two-stream architecture, Convolutional neural networks, Stacked denoising auto-encoder, Class-specific fusion, Deep learning, Remote sensing.**

I. INTRODUCTION

COMPARED with natural color images, hyperspectral images (HSI) contain more channels, and can provide not only more detailed spectral information but also coarse-grained spatial context information. When the spatial context is involved in the classification task, the accuracy can be improved significantly. Therefore, a growing number of scientists have taken into account the spatial context information for hyperspectral image classification [1]. The realted works mainly focus on two directions: 1) extraction of more robust and invariant features, such as hand-crafted features and deep features [2], [3]. 2) fusion of the spectral and spatial features from the complex HSI for classification.

In spatial-spectral classification, the spatial feature extraction is an important problem to be addressed. The strategies for extracting spatial features can be generally divided into the following groups [4], [5]:

- **Neighborhood window**: Spatial features of each pixel can be calculated by averaging all the pixels in the neighborhood window.
- **Segmentation**: The parcels from a segmentation map can be considerd as the homogeneous neighborhood, and can be employed for the extraction of spatial context information. For instance, Tarabalka *et al.* performed segmentation to define the spatial structures [6].
- **Markov random field (MRF)**: The spatial term in the energy function of MRF is used for feature extraction, and MRF can well characterize the relationship between local and global properties of an image [7].
- **Morphological and texture features**: Morphological profiles (MPs) and Gabor filter are the most commonly used and promising methods to characterize spatial context information. MPs were first introduced for hyperspectral images as described in [8]. Then, MPs were further improved with the extended morphological profiles (EMPs) [9]. The attribute profiles (APs) and the extended multi-attribute profiles (EMAPs) are also efficient to represent the context information [10]. Shi and Healey employed a group of Gabor filters to extract spatial features at different scales and orientations from HSI obtaining higher classification accuracy [11].
- **Others**: Apart from the above four groups, other descriptors have also been proposed in the literature. For example, pairwise hyperspectral angle (PHA) can simultaneously explore the context information across the neighboring bands and neighboring pixels [12].

However, all the methods mentioned above can only derive **hand-crafted features** instead of **deep features**, which can be extracted using deep learning networks. The deep features, if properly extracted, can be more discriminative than handcrafted features for the problem of hyperspectral image classification.

During last decade, the representative works in [13], [14] have brought revolution to deep learning methods and have also promoted their development in the field of machine learning and pattern recognition. In addition, deep learning methods have demonstrated the great potential in the field of remote sensing for extracting features, which are abstract, robust and invariant to most local changes of the input values [15]. For instance, Aptoula *et al.* extracted the attribute profiles from the raw data and then gave them as input to convolutional neural networks. In this way, the handcrafted

Manuscript received Nov.19, 2016; revised Dec. 26, 2016. (*Corresponding author: Yuanxin Ye.*)

Siyuan Hao and Tingyuan Nie were with the College of Communication and Electronic Engineering, Qingdao University of Technology, Qingdao 266520, China. E-mail: lemonbananan@163.com, tynie@qut.edu.cn.

Lorenzo Bruzzone and Wei Wang were with the Department of Information Engineering and Computer Science, University of Trento, Italy. E-mail: lorenzo.bruzzone@ing.unitn.it, wei.wang@unitn.it

Yuanxin Ye was with the Faculty of Geosciences and Environmental Engineering, Southwest Jiaotong University, Chengdu 610031, China. E-mail: yeyuanxin@home.swjtu.edu.cn

features can be further enhanced by the convolutional neural networks and better serve the classification task [16]. Chen *et al.* employed the stacked autoencoders to extract the high-level semantic features. They also proved that the semantic features outperformed the hand-crafted features for HSI classification [2]. After that, more researchers started to add the spatial information in deep learning architectures. Thus, a spatial deep auto-encoder, which integrated a similarity regularization term into the loss function, was proposed to extract discriminative spatial-spectral features [17]. Han *et al.* proposed to efficiently represent the spatial-spectral features using the unsupervised convolutional sparse auto-encoder [18]. However, the training computational cost for these methods is high. To address this problem, Romero *et al.* employed deep unsupervised convolutional networks to compute sparse feature representations. They also revealed that the deep convolutional networks are good at extracting discriminative high-level semantic features [19]. A two-stream convolutional network was first presented for action recognition [20]. Yang *et al.* applied this two-channel CNN to the hyperspectral image classification task. However, the input of their spectral CNN channel is vector-based, which means that all the convolutional and pooling operators in this channel work on 1-D features. Besides, they do not train the network from scratch. Their parameters of the bottom and middle layers are transferred from an auxiliary domain [21]. In general, if the network can be trained from scratch, the extracted features can be task-specific and thus serve better the hyperspectral image task [22].

After feature extraction, fusion is another important step for the classification task [23], [24]. The conventional fusion is implemented either on feature-level (early fusion) or on decision score-level (late fusion) [25]. The fusion in [26] was based on the feature level, where the deep features were fused by stacking them directly. 3D convolutional neural networks (3D-CNNs) also implemented the fusion at the feature-level by processing the spectral and spatial features at the same time [27], [28]. For the late fusion, the most common strategy is the uniform weight fusion, which assigns uniform weight values to each probability matrix. For example, Li *et al.* performed the uniform weight fusion on the probability outputs of each classifier. However, they extracted the hand-crafted spatial features by LBP or Gabor filters [29]. Yang *et al.* obtained the probability matrices using two channels and considered them as the corresponding spectral and spatial deep features. Then they fed the stacked feature into the fully-connected layer [21]. However, in term of each classifier, this fusion strategy assumes that the class features make equivalent contributions and the fusion weights share a fixed uniform value. Hence, it fails to take into account the difference of the discrimination capabilities of the features, and the correlation among the features is also ignored.

In order to solve the problems mentioned above, we propose a two-stream deep architecture, which contains a class-specific adaptive fusion scheme. In the two-stream architecture, the first stream (*spectral* stream) is a stacked denoising auto-encoder that is used to encode the spectral features, while the other stream (*spatial* stream) consists of a deep convolutional neural network that is employed to extract deep spatial features. In the fusion scheme, the prediction probabilities from two streams are aggregated by the class-specific fusion.

Compared with the original two-stream architecture in [20], the main of the proposed method: (1) *It is based on a different optimized architecture:* the architecture in [20] only contains ConvNets. Our two-stream architecture contains not only CNNs, but also SdAE to extract the spatial and spectral features. Furthermore, in our work, we employ a softmax layer for classification which is integrated together with the two-stream architecture and the classification layer and the two-stream architecture can be trained jointly in an end-to-end manner. (2) *It is optimized to address the HSI classification task:* our method is proposed for the HIS classification task, while the architecture is designed for the task of action recognition in videos in [20]. In the latter, the inputs are the RGB video frames and the stacked optical flow images, whereas we take as input the spatial image patches and pixel-wise spectral values. As there are plenty of training samples in action recognition task, in [20] authors employed very deep architectures, such as VGG. However, the training samples are very limited in HIS classification task. Therefore, we designed an appropriate light CNNs for the HIS classification task.

The proposed architecture has the following advantages:

(1) Deep spectral and spatial features can be extracted by each stream, which can be trained from scratch. Therefore, the deep features can be learned specifically for the classification task.
(2) A class-specific fusion scheme is presented to learn the class fusion weights adaptively taking into account the correlation among different classes.
(3) To alleviate the overfitting of the fusion weight values, the loss function is improved by adding a weight regularizer.

The rest of the paper is organized as follows. Section II briefly reviews the background on deep learning architectures. Section III introduces the proposed method. Section IV evaluates the effectiveness of the proposed method using two hyperspectral datasets. Section V analyzes the experimental results and draws the conclusion of the paper.

II. Background on deep learning architectures

A. Denoising Auto-encoder (DAE)

An auto-encoder (AE) is a multi-layer neural network with one input layer, one output layer and one hidden layer. AE first takes as input a signal and maps it into the hidden layer through a deterministic mapping function. Then the output layer reconstructs the signal by minimizing the reconstruction error. Thus, more complex and abstract nonlinear representations can be hierarchically learned from the hidden layer. The mapping function could either be a nonlinear function (*e.g.*, sigmoid, ReLU) or a linear function. When a linear mapping function is applied, the auto-encoder will behave like a PCA.

A denoising auto-encoder (DAE) [30] is an extension of the standard AE with a similar structure as shown in Fig.1. There are two steps in DAE, *i.e.,* encoding and decoding, both of which are implemented by nonlinear mappings. However, differently from AE, the input of DAE is corrupted by adding noise, while the output is the original signal without noise.

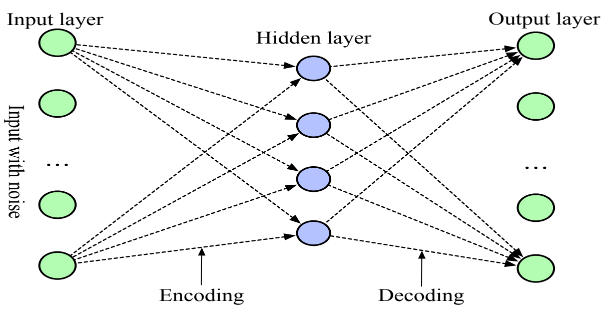

Fig. 1. **Structure of denoising auto-encoder.**

In this way, the learnd DAE is capable of recovering signal from the noisy input. The learnt representation via DAE is more robust than that of AE. Therefore, DAE has received significant amount of attention in the research community, especially for speech recognition. For example, Ueda *et al.* proposed a DAE variant for robust distant-talking speech recognition [31]. Lu *et al.* presented a model by ensembling DAEs, which effectively focuses on local transformations [32]. S. Araki *et al.* introduced a multi-channel DAE-based speech enhancement approach [33]. DAE model can be also applied to human pose-based action recognition [34].

Recently, deep learning has also been successively applied to the field of remote sensing. The remote sensing scientists use AE/DAE to extract features for subsequent classification and unmixing [2], [35]–[37]. In [2], AE was used to extract features for classification, whilst Chen *et al.* employed multiple DAEs to extract deep features [36]. The autoencoder cascade was presented in [37], which concatenated a marginalized denoising autoencoder and a non-negative sparse autoencoder for unmixing.

B. Stacked Denoising Auto-encoder (SdAE)

Several DAEs can be stacked into a deep network by feeding the output from the hidden layer of the previous DAE as the input to the next DAE. This deep network is named as the stacked denoising auto-encoder (SdAE). The SdAE is usually used to extract high-level representation of input in the noisy scenarios, and then SdAE is fine-tuned for the subsequent classification task. The training process can be divided into two stages: unsupervised pre-training of each DAE and supervised fine-tuning for the whole SdAE. In the stage of pre-training, the greedy layer-wise training is used for each DAE. In the stage of fine-tuning, the bottom layers of the SdAE are first initialized by the pre-trained parameters. Next, a logistic regression layer is added on the top of the network. Then, the network is further tuned by the logistic regression layer. By minimizing the classification error from the logistic regression layer, the deep semantic features can be learned.

SdAE has been widely used in many applications [38]–[42]. For example, SdAE was exploited to improve the accuracy of transfer learning in [39]. Glorot *et al.* applied SdAE for the sentiment analysis task and demonstrated that these deep features, when combined with a simple linear SVM classifier, yielded the state-of-the-art performance outperforming the hand-crafted features [43]. Subsequently, some variants of SdAE were also proposed. For instance, the marginalized denoising autoencoder was presented, which marginalized the random corruption to yield the optimal reconstruction weights, and these weights were computed in a closed-form [42]. This method does not use back-propagation for the parameter tuning, and it results in a relatively low training cost. The marginalized SdAE was further combined with probabilistic matrix factorization to learn the sparse representation for the collaborative filtering [40]. For addressing the hyperspectral image classification task, the methods in [41], [44] used the spectral representation extracted by SdAE instead of the raw data, which can well reflect the nonlinearity in a high dimensional data space.

C. Convolutional Neural Networks (CNNs)

The Convolutional Neural Networks (CNNs) are biologically inspired variants of Multi-layer Perceptions. The most widely used layers in CNNs are vision layers, activation layers and loss layers. A representive structure of CNNs is shown in Fig.2. In the vision layers, the image patches with spatial context information are taken as the input. They are first convolved with a set of kernels with the same size, and then the pooling operation is carried out. The activation function is implemented to learn the nonlinear representations [19]. The activation operators are usually element-wise, such as sigmoid, ReLU and hyperbolic functions. In the loss layer, the most widely used loss is the softmax cross-entropy loss.

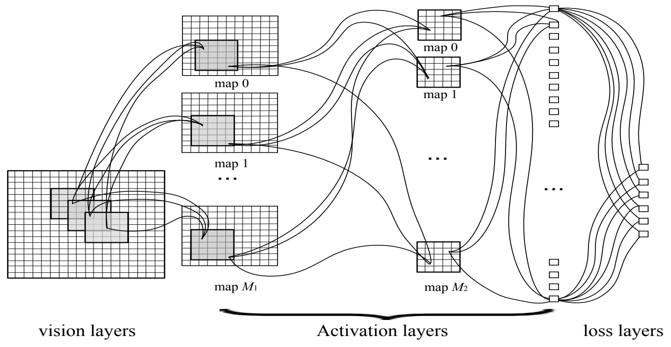

Fig. 2. **Structure of convolutional neural networks.**

The latest researches show that features extracted by a deeper network are more abstract and thus have better performances in the classification task. Therefore, many deeper convolutional networks have been constructed in recent years, such as AlexNet (8 layers) [45], VGG-Net (16-19 layers) [46], GoogLeNet (22 layers) [47] and residual net (152 layers) [48]. These deep networks require lots of training data to learn the parameters. However, the collection for the labeled training data of hyperspectral images is very laborious and expensive, and the limited training data will cause the over-fitting problem. Therefore, even if these deeper networks have a significant advantage in extracting features, shallow and small networks are more appropiate for hyperspectral images. This problem will be further discussed in Section IV.

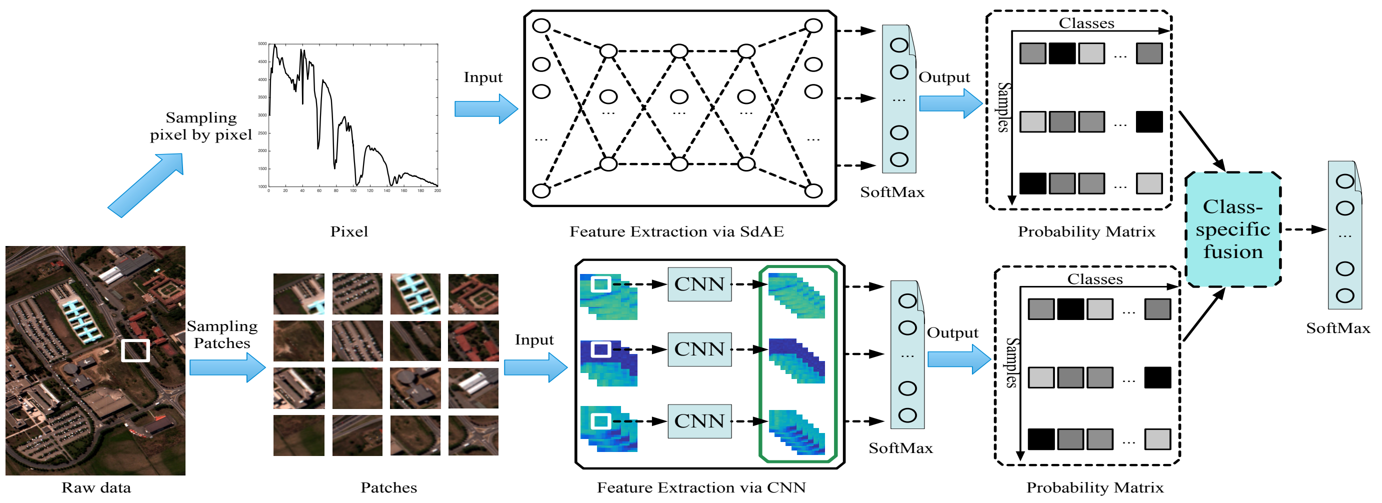

Fig. 3. **Overview of the proposed two-stream deep architecture. The stream on the top is the spectral stream which is a stacked denoising auto-encoder. The input for this stream are the spectral pixels. The bottom stream is the spatial stream which is a deep convolutional neural network. The input for the spatial stream are the image patches.**

CNNs have shown an explosive popularity in image classification [45], [49]. For instance, Ciresan *et al.* presented a fast, fully parameterizable GPU implementation of CNNs variants for image classification. It has very good generalization capability and high speed. However, it is computationally prohibitive on CPUs [50]. In order to carry out CNNs on CPUs with great efficiency, an end-to-end CNNs architecture and a 3-dimension CNN-based feature extraction model were presented in [51] and [52], respectively. These models can boost the discriminative capability and robustness of the extraced features. In addition, a mixed deep learning model composed by both deep auto-encoder and CNNs was proposed [22]. It first employed a pre-trained CNNs from an auxiliary domain (RGB images) for feature extraction. Then a deep auto-encoder was used to process the deep features. These two steps were implemented independently. Thus, pre-trained CNNs have a poor adaptive capability for HSI datasets since they are neither trained or fine-tuned using HSIs.

III. Proposed Architecture

A major challenge in hyperspectral image classification is how to fully and efficiently extract and fuse the spectral information of a single pixel and its spatial correlations with the neighbouring pixels in the same local context. In order to address this issue, we propose a two-stream deep architecture which incorporates and fuses spectral and spatial information. As shown in Fig.3, the proposed two-stream deep architecture consisits of two streams (*i.e.* the spectral stream and the spatial stream) and a class-specific fusion architecture. The spectral stream is a SdAE that handles the spectral pixels, while the spatial stream is composed of CNNs which are in charge of processing the spatial image patches. The class-specific fusion scheme can learn the fusion weight values of different classes adaptively, and these weights can be further optimized in the training process.

A. Spectral stream

DAE is the basic component for SdAE. Given a hyperspectral image pixel $\mathbf{x} \in \mathbb{R}^d$ represented in d-dimension, it should be first corrupted into $\tilde{\mathbf{x}}$ by setting some elements of $\mathbf{x}$ to zeros or adding a Gaussian noise before feeding it into DAE. The hidden representation of $\tilde{\mathbf{x}}$ is computed by:

$$\mathbf{h} = s(\mathbf{W}\tilde{\mathbf{x}} + \mathbf{b}), \tag{1}$$

where $\mathbf{h} \in \mathbb{R}^{d_1}$, and d_1 is the number of units in the hidden layer. $\mathbf{W}$ and $\mathbf{b}$ are the encoding weight and the corresponding bias vector, respectively. s is the activation function (*e.g.,* the sigmoid, ReLU). We use the ReLU in this stream. The hidden representation is then mapped back to reconstruct $\mathbf{z}$ without noise using the following function:

$$\mathbf{z} = s(\mathbf{W}'\mathbf{h} + \mathbf{b}') \tag{2}$$

where $\mathbf{W}'$ and $\mathbf{b}'$ are the decoding weight and bias vector. The reconstruction error can be measured in many ways, depending on the distribution assumptions on the input dataset. In this paper, we choose cross-entropy to compute the reconstruction error, and minimize it to learn the optimal parameter $\Theta^* = \{\mathbf{W}, \mathbf{W}', \mathbf{b}, \mathbf{b}'\}$ by:

$$\Theta^* = \min_{\Theta} - \sum_{k=1}^{d} [\mathbf{x}_k \log \mathbf{z}_k + (1 - \mathbf{x}_k)\log(1 - \mathbf{z}_k)] \tag{3}$$

DAEs can be stacked to form SdAE by feeding the hidden representations of the previous DAE layer as the input to the current DAE layer. The spectral stream (on the top row in Fig. 3) analyzes the spectral similarity encoding by SdAE, which contains one input layer, three DAE layers and one output layer. For parameter optimization, we train the first DAE layer (encoding and decoding) and obtain the parameters Θ_1, and then feed the hidden representation of the first DAE layer to the second DAE layer. The remaining two DAE layers can be trained in the same manner and the parameters Θ_2, Θ_3 can be learned.

Different from other works in which the feature extraction (*e.g.*, HOG) and classifier (*e.g.*, SVM) training processes are implemented separately, the proposed spectral stream can be fine-tuned in an end-to-end manner using the semantic labels as supervision. The labels of test set can be immediately predicted by the softmax layer. Therefore, the efficiency is higher than those of other one-versus-all classifiers.

B. Spatial stream

As shown in the bottom row of Fig.3, the spatial stream is CNNs which are used to extract spatial features. The convolutional layer generates the feature maps by convolving the output (*i.e.,* feature maps) of its previous layer with a set of kernels. Then, the pooling layer is applied to obtain more abstract and transition invariant features. Pooling operator can greatly reduce the computational cost and increase the generalization ability of the model [53].

In order to take advantage of the spatial context information, image *patches* centered at the corresponding pixels from the spectral stream are cropped and fed into CNNs. Since the number of these labelled image patches is usually quite small compared with that of the available natural images databases, we build a shallow CNN architecture with two convolutional layers to process these image patches. A more detailed study of the optimal number of convolutional layers is available in Section IV-D.

Let $\mathbf{x} \in \mathbb{R}^d$ represent the hyperspectral image pixel with d-dimension, and $\mathbf{S} \in \mathbb{R}^{d\times m\times m}$ denote the image patch centered at $\mathbf{x}$, where the spatial size of the image patch at each channel is $m \times m$. The kernel size for the two convolutional layers is set to $[k_1, k_2]$ and their kernel numbers are set to n_1 and n_2, respectively. The kernel size for two pooling layers is set to $[s_1, s_2]$. Formally, the first layer can be expressed as an operation F_1:

$$F_1(\mathbf{S}) = \max(0, \mathbf{W}_1 * \mathbf{S} + \mathbf{B}_1) \tag{4}$$

where $\mathbf{W}_1$ and $\mathbf{B}_1$ denote the kernels and biases, and '$*$' denotes the convolutional operation. Here, $\mathbf{W}_1$ corresponds to n_1 kernels with size $d \times k_1 \times k_2$. The output of the first layer is composed of n_1 feature maps.

The convolutional layer reduces the image size to $[m-k_1+1, m-k_2+1]$, and the pooling layer further reduces the feature maps to the size of $[\frac{m-k_1+1}{s_1}, \frac{m-k_2+1}{s_2}]$. The operation of the second convolutional layer is:

$$F_2(\mathbf{S}) = \max(0, \mathbf{W}_2 * F_1(\mathbf{S}) + \mathbf{B}_2) \tag{5}$$

where $\mathbf{W}_2$ contains n_2 kernels having size $n_1 \times k_1 \times k_2$, and $\mathbf{B}_2$ has a dimension equal to n_2.

Then the image size is reduced to $[f_1, f_2]$, and $f_1 = (\frac{m-k_1+1}{s_1} - k_1 + 1)/s_1$, $f_2 = (\frac{m-k_2+1}{s_2} - k_2 + 1)/s_2$. For the fully-connected layer, the feature maps obtained from the previous layer should be reshaped to a 2-dimension matrix, because the fully-connected layer can only operate on 2-dimension matrices. The used expression is as follows:

$$F_3(\mathbf{S}) = s(\mathbf{W}_3 F_2(\mathbf{S}) + \mathbf{B}_3) \tag{6}$$

where $\mathbf{W}_3$ is a matrix with the size $[n_2 \times f_1 \times f_2, n_3]$. $\mathbf{B}_3$ is a n_3-dimensional vector, where n_3 is the number of hidden units of the fully-connected layer.

Similar to the spectral stream, the spatial stream can also adjust the parameters depending on the distribution of the input data, and works in an end-to-end manner. It is worth noting that the two streams can be fine-tuned jointly, and the features from the two streams can be efficiently taking into account their different informative content and correlation. This can benefit the overall classification task.

C. Class-specific Fusion

Late fusion is one of the most effective schemes to enhance classification performance through combining prediction scores of multiple classifiers, each of which can be trained by a specific type of features. The proposed two streams are implemented using the spectral SdAE and spatial CNNs, and each stream can directly obtain the predicted probability score with respect to the input. Therefore, a robust prediction and better performance can be achieved by fusing the different classification results.

Differently from the traditional uniform weight fusion, we attempt to learn the fusion weight values of different classes adaptively when training the whole network. Let $\mathbf{p}_1$ and $\mathbf{p}_2 \in \mathbb{R}^{1\times N'}$ represent the probability vectors obtained from the spectral and spatial streams, and N' be the number of classes. The correspongding fusion weight is represented by $\omega_i = \{\omega_i^1, \omega_i^2, ..., \omega_i^j, ..., \omega_i^{N'}\}(i \in 1, 2)$ where ω_i^j is the j-th class fusion weight of the i-th stream. We can express the fused vector as follows:

$$\begin{aligned}
&\mathbf{p} \cdot \omega = \mathbf{p}_1 \odot \omega_1 + \mathbf{p}_2 \odot \omega_2 \\
&= [\mathbf{p}_1^1, \cdots, \mathbf{p}_1^{N'}, \mathbf{p}_2^1, \cdots, \mathbf{p}_2^{N'}]
\begin{bmatrix}
\omega_1^1 & 0 & \cdots & 0 \\
0 & \omega_1^2 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & \omega_1^{N'} \\
\omega_2^1 & 0 & \cdots & 0 \\
0 & \omega_2^2 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & \omega_2^{N'}
\end{bmatrix}
\end{aligned} \tag{7}$$

where $\mathbf{p}$ is the stacked probability vector formed by $\mathbf{p}_1$ and $\mathbf{p}_2$, $\omega \in \mathbb{R}^{2N' \times N'}$ is a matrix composed of the class fusion weights, and $\odot$ represents the element-wise product.

By analyzing (7), we can observe that the target of fusion is to learn the optimal class fusion weight values, which is equivalent to the formula of learning the weights of a fully-connected layer. To this end, we use a fully-connected layer to construct the class-specific fusion architecture. A softmax layer is added on the top of the class-specific fusion architecture. In this way, the class fusion weights can be learnt and adjusted adaptively via minimizing the loss function of this softmax layer. Instead of assigning random values to the weights of the fully-connected layer, we only initialize the elements on the diagonal while the other elements are simultaneously set to 0. This initial constraint will be lossen during the training process, which means that the correlation among different classes can also be taken into account in the fusion process.

To alleviate the risk of overfitting of the fusion weights, a regularizer is integrated into the loss function of the softmax layer. The overall loss function is given as follows:

$$\omega = \min_{\omega} loss(\mathbf{p}, y; \omega) + \|\omega\|_2 \tag{8}$$

where $loss(\cdot)$ is the original loss function of softmax layer, y represents the ground-truth labels, and $\|\cdot\|_2$ is the l_2 norm.

The conventional fusion is always independent from the feature learning process, and the fusion weights are set manually. Differently, in our work, the feature fusion is integrated together with the feature learning pipeline. Therefore, the fusion weights can be learned adaptively by referring to the features that are learned and the features can also be tuned by referring to the fusion weights. In this way, the joint training of the overall architecture can boost the HIS classification performance.

IV. Experiments

A. Datasets description and parameter setting

The experiments to evaluate the effectiveness of the proposed two-stream deep architecture have been conducted on two hyperspectral images.

The first hyperspectral image is the Indian Pines. It contains 220 spectral bands and has a size of 145×145 pixels, with each pixel measuring approximately 20m by 20m on the ground. 20 spectral bands are removed due to the noise and water absorption phenomena. Sixteen mutually exclusive classes are included. Some classes in this dataset have very few samples, thus we split the dataset into training and testing sets with the ratio of 3:2. The three-channel false-color composition and the reference land-cover map of Indian Pines are shown in Fig. 4.

The second hyperspectral image is the University of Pavia. Water absorption bands are removed, and the original 115 bands are reduced to 103 bands. Fig.5 shows the three-channel false-color composition and the reference land-cover map of University of Pavia, which mainly includes 9 classes (*i.e.*, asphalt, meadows, trees, metal sheets, bare soil, bitumen, bricks, shadows and gravel). For this dataset, we set the ratio of the training set and testing set to 1:9.

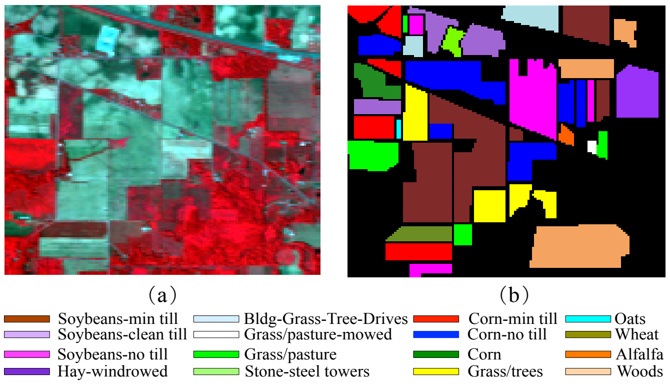

Fig. 4. **Indian Pines dataset: (a) three-channel false-color composition (bands 17, 27, and 50 for RGB) (b) reference land-cover.**

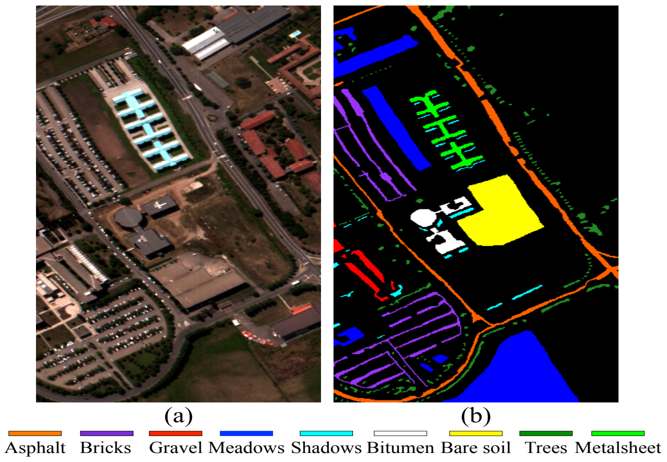

Fig. 5. **Pavia University dataset: (a) three-channel false-color composition (bands 16, 27, and 45 for RGB) (b) reference land-cover.**

For the spectral stream, the network weights are learnt using the mini-batch Stochastic Gradient Descent (SGD) with respect to the loss defined in (3). SdAE was composed of three DAE layers, in which the number of hidden units for each DAE layer was set to 10^2. On the top of the spectral stream, a softmax layer was added to predict the scores of samples. The corruption level was selected from the set of $[0.1, 0.2, 0.3]$, because larger values may lead to the overfitting problem. Following [2], the number of epochs for pre-training and fine-tuning were both set to 10^3. In the process of SGD, the *initial* learning rates in the pre-training and fine-tuning phases were both set to 10^{-1}, and the weight decay was 0.5. The learning rates are decreased every 10^2 iterations, and become almost one thousandth of the initial value when pre-training and fine-tuning are terminated. For the spatial stream, two convolutional layers, one fully-connected layer and one softmax layer were stacked to construct the CNNs model, in which the number of hidden units in fully-connected layer was 10^2, and the softmax layer was used to predict the scores of samples by the deep sptial features. For the class-specific fusion architecture, the initialization of the weights in the fully-connected layer is described in Section III-C.

The performance evaluation metrics are the overall accuracy (OA), the average accuracy (AA), the F1-score and the Precision. To avoid biased estimation, ten independent tests were carried out using Theano and Keras on a computer equipped with an Intel Core i5 Processor at 2.70-GHz.

B. Analysis of Class-specific fusion

Fig.6 shows the evolution of the class fusion weight values on the dataset of the University of Pavia. The fusion weight matrix is composed of ω_1 for the spectral stream (on the top) and ω_2 for the spatial stream (at the bottom). The size of the fusion weight matrix for the considered dataset is 18×9. As stated in Section III-C, we only assigned the initial values of 0.5 to the diagonal of the weight matrix of the fully-connected layer (see Fig.6 (a)). In the testing phase, we selected 10% samples from each class as the training set. We can observe that as the number of training epochs increases the values in the fusion weight matrix also evolve accordingly. Comparing the values in the first column (which corresponds to the fusion weights of the first class) of the two weight matrices, one can see that at the beginning $\omega_1^1 = \omega_2^1 = 0.5$ (Fig.6 (a)), while after 1000 training epochs ω_1^1 and ω_2^1 decrease to 0.4082 and 0.429 (Fig.6 (b)), respectively. At the same time, the value located at (*row 4, column 1*) increases to 0.2919, which means that the features of the fourth class in the spectral stream greatly contribute to the classification of the samples from the first class. Similar behaviours also take place in other columns. The observation of the fusion weight demonstrates that in the training phase the class-specific fusion scheme can adaptively tune the weights of each class by taking into account the correlations between them.

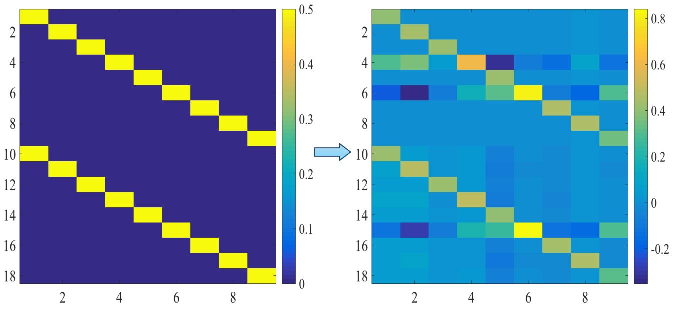

Fig. 6. **Evolution of the class fusion weights for Pavia University dataset: (a) initial weights, (b) weights obtained after 1000 training epochs.**

C. Analysis of the performance of DAE

In order to prove that the spectral features extracted by DAE are more robust, we implemented the following test using the Indian Pines dataset. We randomly selected 60% and 20% of the total samples as the training and validation sets to learn the parameters $\Theta^* = \{\mathbf{W}, \mathbf{W}', \mathbf{b}, \mathbf{b}'\}$, and the remaining ones were used as the test set. $\mathbf{W}'$ was constrained such that $\mathbf{W}' = \mathbf{W}^T$, and $\mathbf{W}$ was initialized by uniformly sampling from the range $[-0.5, 0.5]$. The reconstructed signals by AE (the middle column in blue) and DAE (the last column in red) after 500 and 1000 training epochs are shown in Fig.7.

The first row in Fig. 7 shows the ground truth curve, and the reconstructed spectral curves of AE and DAE after 500 epochs. We can observe that the reconstructed signals cannot completely match with the original one with a limited number of training epochs. The curves obtained by AE and DAE are similar, and DAE has the capability to reconstruct signal under noisy conditions. Then, we compare the results in each column. We can observe that the performance of DAE is improved when the number of the training epochs increases from 500 to 1000. By analyzing the second row we can observe that the performance of DAE (last column) is better than that of AE (middle column), which can be noted in the spectral bands between [0, 25] and [100, 200]. The spectral signals located in these spectral bands increase significantly and move closer to the original signal. Therefore, we can conclude that DAE can be a good alternative to AE to extract spectral features that are more robust. This is the main reason for which we choose the DAE to stack the multi-layer network.

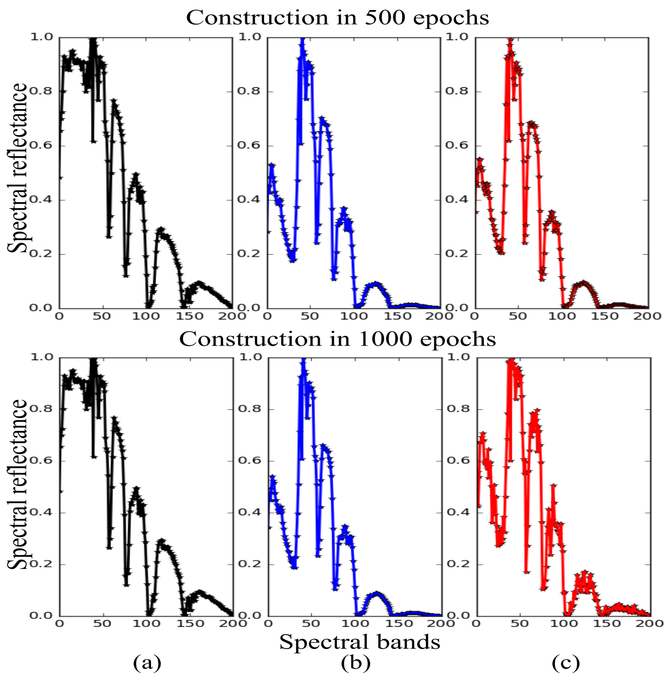

Fig. 7. **Reconstruction of a sample in Indian Pines with different training epochs: (a) a randomly selected test sample, (b) AE, (c) DAE.**

D. Effect of the network depth

The depth of a network plays a very important role. Usually, deeper architecture can extract more abstract features, which is crucial for the classification performance. In this part, we conducted the following two experiments to find out the optimal network depths for the spectral stream and the spatial stream on different datasets.

1) Effect on the spectral stream: The network depth in SdAE corresponds to the number of DAE layers. Considering the high computational cost of training for fully-connected networks, we only evaluated the depth from 1 to 5 layers. The spatial size of the input patches is fixed to 7×7. All the other settings remained the same as in Section IV-A. Then we predicted the labels of the test set, and listed the corresponding classification accuracy for the dataset of Indian Pines and the University of Pavia in Tab. I and Tab. II, respectively.

As shown in Tab. I, we can see that the performance generally shows an upward trend with a deeper network. It reaches the peak values (98.65% OA, 99.22% AA, 98.65% F1-score and 98.68% Precision) with a depth of 3. We can also observe that the accuracy fluctuates when the depth is greater than 3. When the depth is set to 5, the accuracy rises again, but is still inferior compared with the depth of 3. From these observations, we can conclude that a deeper SdAE helps to improve the performance. However, after a given value, increasing the depth decreases the performance because of the problem of overfitting. A 3-layer SdAE is good enough for the Indian Pines dataset. From Tab. II, we can observe that a 4-layer SdAE is better suited to the University of Pavia dataset, and the OA, AA, F1-score and Precision can reach 97.33%, 95.01%, 97.32% and 97.39%, respectively. Therefore, we use a 3-layer SdAE for the Indian Pines and a 4-layer one for the

University of Pavia in the final two-stream deep architecture.

TABLE I
EFFECT OF THE DEPTH ON THE SPECTRAL STREAM (INDIAN PINES).

Depth (layers)	OA	AA	F1-score	Precision
1	86.96	90.22	86.57	87.36
2	94.88	96.09	94.85	94.91
3	**98.65**	**99.22**	**98.65**	**98.68**
4	98.50	99.15	98.50	98.53
5	**98.65**	99.21	**98.65**	**98.68**

TABLE II
EFFECT OF THE DEPTH ON THE SPECTRAL STREAM (PAVIA UNIVERSITY).

Depth (layers)	OA	AA	F1-score	Precision
1	94.95	90.73	94.77	94.99
2	95.74	92.68	95.74	95.80
3	96.20	93.15	96.17	96.24
4	**97.33**	**95.01**	**97.32**	**97.39**
5	96.52	94.69	96.52	96.69

2) Effect on the spatial stream: Similar to the spectral stream, we also used the same dataset to train and test the spatial stream using the different depths. The network depth varied in the range from 1 to 5. We fixed the size of spatial patch to 7×7. Considering the relatively small spatial size of the Indian Pines image (*i.e.,* 145×145 pixels), we removed the pooling operator behind the convolutional layer. For a fair comparison, the kernal size of all the convolutional layers was set to 2×2, and its kernel number was set to 50. The classification results for the two datasets are summarized in Tab. III and Tab. IV.

From Tab. III, we can derive the following observations: 1) The OA generally increases until $depth = 3$, reaching the peak of 98.65%. Then, OA fluctuates slightly as the number of layers increases further. 2) CNNs with three layers can produce the largest AA, which suggests that a deeper CNNs model can well learn the distribution of classes. 3) The optimal F1-socre and Precision can be achieved at $depth = 3$. Therefore, we select CNNs with three layers for the Indian Pines dataset in the following tests. In addition, we can also observe from Tab. IV that the highest OA, AA and recall can be obtained when $depth = 2$ for the University of Pavia dataset.

Obviously, each stream is heavily influenced by the network depth and the selection of network depth for each stream plays an important role in the performance obtained.

TABLE III
EFFECT OF THE DEPTH ON THE SPATIAL STREAM (INDIAN PINES).

Depth (layers)	OA	AA	F1-score	Precision
1	97.92	98.56	97.92	97.94
2	97.54	98.81	97.54	97.61
3	**98.65**	**99.22**	**98.65**	**98.68**
4	98.31	98.95	98.31	98.33
5	97.78	97.43	97.79	97.85

E. Effect of the spatial size

The spatial size s of the input image patches is another important parameter for the proposed two-stream deep architecture. In this section, we compare the classification accuracy (OA) of different spatial sizes with different amount of training data. For the Indian Pines dataset, we randomly selected training data from the whole dataset with a ratio r, which was set to $10\%, 20\%, 30\%, 40\%, 50\%, 60\%$. The ratio r of the Pavia University dataset varied in the set $\{1\%, 3\%, 5\%, 10\%, 15\%, 20\%\}$. The spatial size varied in the set $\{3\times3, 5\times5, 7\times7\}$, and the kernel size of two convolutional layers was 2×2. For the small spatial size (*i.e.*, 3×3), we removed the pooling layer because of the rapid downsampling of image size. The OA curves are shown in Fig.8.

TABLE IV
EFFECT OF THE DEPTH ON THE SPATIAL STREAM (PAVIA UNIVERSITY).

Depth (layers)	OA	AA	F1-score	Precision
1	93.16	85.46	92.46	93.23
2	**97.33**	**95.01**	**97.39**	**97.39**
3	96.24	93.99	96.21	96.30
4	95.32	93.36	95.37	95.37
5	95.92	93.91	95.92	95.94

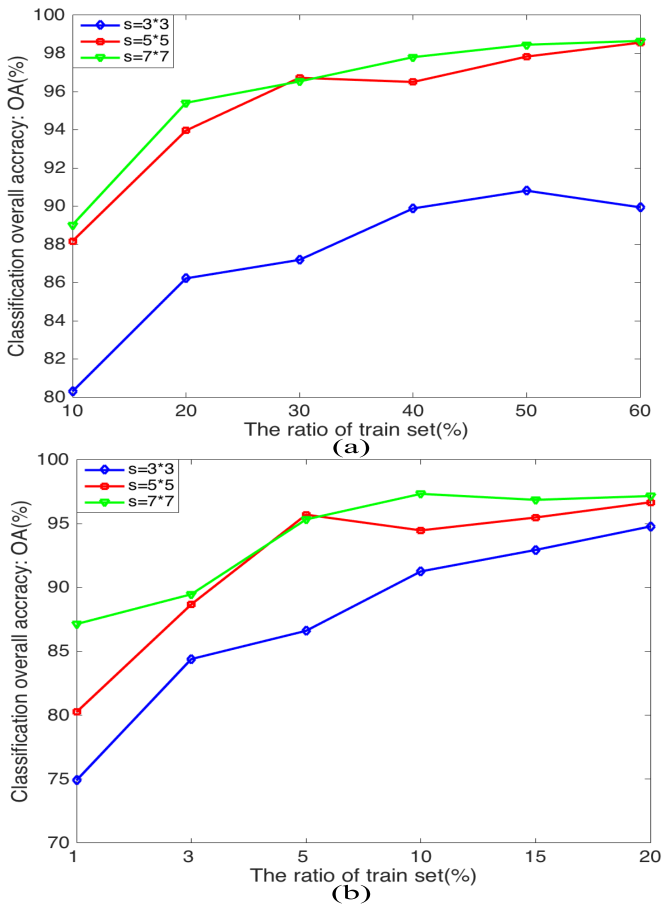

Fig. 8. **Accuracy curves for different spatial sizes versus different training sample ratios: (a) Indian Pines dataset (b) Pavia University dataset.**

From Fig. 8 (a), one can observe that the curves of $s = 7 \times 7$ and $s = 5 \times 5$ have a similar performance improvement as the training set ratio increases. More specifically, the OA curve of $s = 7 \times 7$ increases monotonically from 89.02% ($r = 10\%$) to 98.65% ($r = 60\%$). When r increases from

TABLE V
COMPARISON OF CLASSIFICATION ACCURACIES PROVIDED BY DIFFERENT METHODS (INDIAN PINES DATASET).

Class	No. of train	SS-LPSVM	SC^{3}SVM	SdAE	CNNs	3D-CNNs	ppMLR	MLRpr	ppMLRpr	Proposed
Alfalfa	33	98.89	91.98	**100.00**	**100.00**	96.67	98.15	96.30	98.15	**100.00**
Corn-no till	861	75.45	52.67	73.91	92.61	94.62	**98.47**	80.33	95.12	95.35
Corn-min till	501	75.08	68.15	72.97	90.81	98.68	98.44	68.11	97.48	**98.75**
Corn	141	95.30	78.63	88.46	94.23	97.75	97.86	**100.00**	99.15	**100.00**
Grass/pasture	299	93.40	90.48	91.53	96.61	98.54	95.17	93.16	95.98	**100.00**
Grass/trees	449	95.72	91.48	93.75	95.83	97.40	99.73	**99.87**	99.60	99.32
Grass/pasture-mowed	9	95.38	89.74	**100.00**	**100.00**	**100.00**	**100.00**	42.31	57.69	**100.00**
Hay-windowed	294	97.22	87.46	**100.00**	**100.00**	99.45	**100.00**	**100.00**	**100.00**	**100.00**
Oats	12	**100.00**	95.00	**100.00**	**100.00**	87.50	95.00	0.00	0.00	**100.00**
Soybeans-no till	580	82.77	71.01	87.96	95.81	97.86	96.49	75.83	90.60	**100.00**
Soybeans-min till	1480	65.37	56.39	88.25	96.91	98.43	98.46	96.27	**99.47**	98.03
Soybeans-clean till	369	87.30	73.40	62.42	85.23	95.65	98.53	94.79	99.19	**100.00**
Wheat	127	99.62	99.21	97.44	**100.00**	**100.00**	99.53	**100.00**	99.06	97.87
Woods	777	97.74	92.30	98.25	99.30	99.63	99.77	99.77	**99.85**	99.62
Bldg-Grass-Tree-Drives	228	88.89	62.81	80.00	96.67	96.64	99.21	75.26	**99.74**	98.53
Stone-steel towers	57	96.84	95.09	**100.00**	**100.00**	**100.00**	**100.00**	97.89	93.68	**100.00**
AA	-	90.31	80.99	89.68	96.50	97.43	98.43	82.49	89.05	**99.22**
OA	-	84.11	72.42	86.04	95.36	97.77	98.50	89.61	97.41	**98.65**

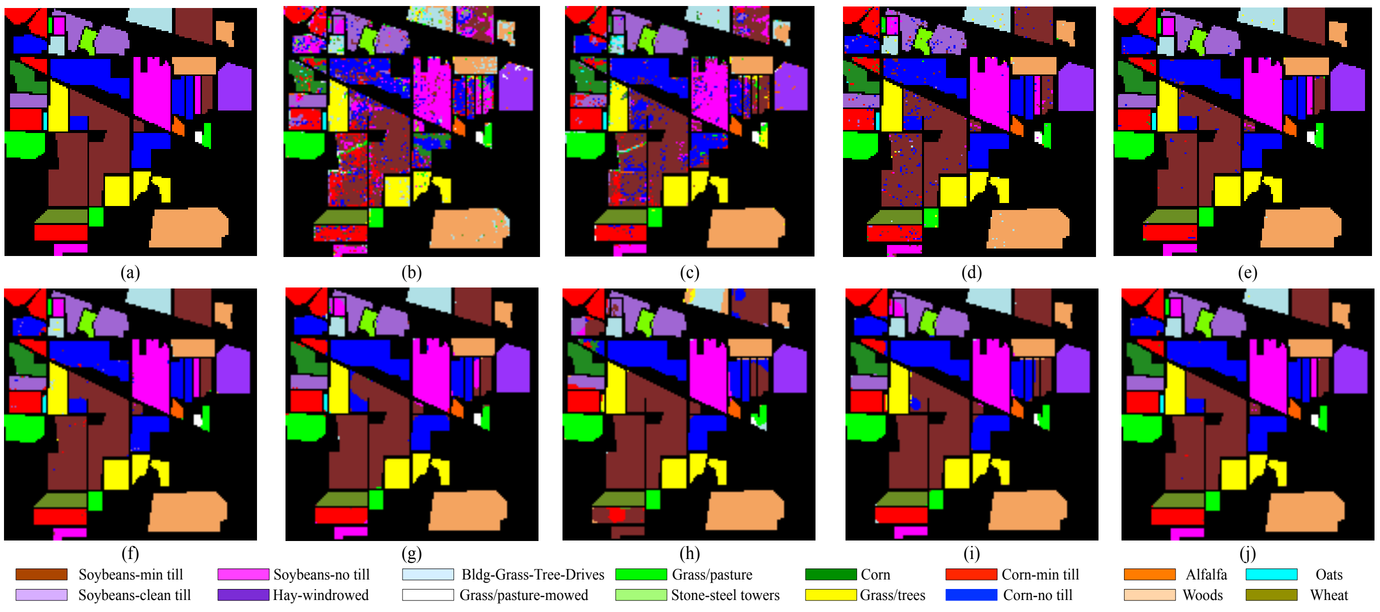

Fig. 9. **Visual classification results for the Indian Pines dataset: (a) reference land-cover (b) SC^{3}SVM (c) SS-LPSVM (d) SdAE (e) CNNs (f) 3D-CNNs (g) ppMLR (h) MLRpr (i) ppMLRpr (j) proposed method.**

10% to 60%, the OA curve of $s = 5 \times 5$ raises 10.39% and a slight performance drop takes place at $r = 40\%$. As expected, the curve of $s = 3 \times 3$ first increases until $r = 50\%$, and then decreases linearly. If we fix the training ratio $r = 40\%$, we can observe that the OA curve increases rapidly by increasing spatial size. It reaches the peak of 97.80% at $r = 40\%$, and the improvement is more relevant from $s = 3 \times 3$ to $s = 5 \times 5$. The behavior of the OA curve for the Pavia University dataset (Fig. 8(b)) is similar to that of the OA curve in Fig. 8(a), that is, the OA curves generally raise when the training set ratio increases. The curves of $s = 7 \times 7$ and $s = 5 \times 5$ have a significant advantage over the curve of $s = 3 \times 3$. With respect to the curve of $s = 7 \times 7$, the OA first increases from 87.14% ($r = 1\%$) to 97.50% ($r = 10\%$). If the ratio continues to increase, similar OAs are obtained (96.86% at $r = 15\%$). Then a slight fluctuation is observed at $r = 20\%$. Differently from the Indian Pines dataset, when only a small amount of training samples is available, the proposed method can still achieve an accurate performance, *e.g.,* OA at $r = 10\%$ can reach 97.50% for the curve of $s = 7 \times 7$. The suitable spatial size is $s = 7 \times 7$ or $s = 5 \times 5$. We also add the pooling operation behind the convolutional layer since it can bring more nonlinearity to the network model. Thus, $s = 7 \times 7$ will be adopted in the proposed architecture.

F. Comparsion with other methods

Tab. V reports the classification accuracy for different methods when applied to the Indian Pines dataset. The spatial-spectral classification methods (SC^{3}SVM, SS-LPSVM, ppMLR, MLRpr and ppMLRpr) in [54]–[57] focus on the integration of spatial context information in classification for the improvement of accuracy. Therefore, they were selected as

TABLE VI
COMPARISON OF CLASSIFICATION ACCURACIES PROVIDED BY DIFFERENT METHODS (PAVIA UNIVERSITY DATASET).

Class	No. of train	SS-LPSVM	SC^{3}SVM	SdAE	CNNs	3D-CNNs	ppMLR	MLRpr	ppMLRpr	Proposed
Asphalt	664	94.18	55.04	87.32	92.00	94.05	91.71	83.36	91.51	**97.47**
Meadows	1865	76.14	68.50	91.48	97.69	98.44	97.48	94.96	95.04	**99.92**
Gravel	210	76.04	47.93	0.00	74.33	79.42	66.33	52.08	53.15	**83.80**
Trees	307	95.17	61.46	83.87	93.84	97.47	95.52	97.00	95.89	**98.98**
Metal sheets	135	99.55	88.03	97.99	**100.00**	**100.00**	99.82	99.36	98.82	**100.00**
Bare soil	503	68.64	38.34	24.33	83.09	89.36	82.80	81.20	85.61	**97.75**
Bitumen	133	88.05	**93.46**	1.99	78.45	86.67	76.65	72.79	72.88	77.44
Bricks	369	74.31	77.19	81.48	76.17	86.65	82.89	85.38	82.62	**96.65**
Shadows	95	**99.68**	98.84	98.66	95.99	97.76	98.82	98.95	98.82	99.65
AA	-	85.75	69.86	63.01	87.95	92.20	88.00	85.01	86.04	**94.63**
OA	-	80.88	64.16	74.70	91.26	94.35	91.42	88.31	89.85	**97.50**

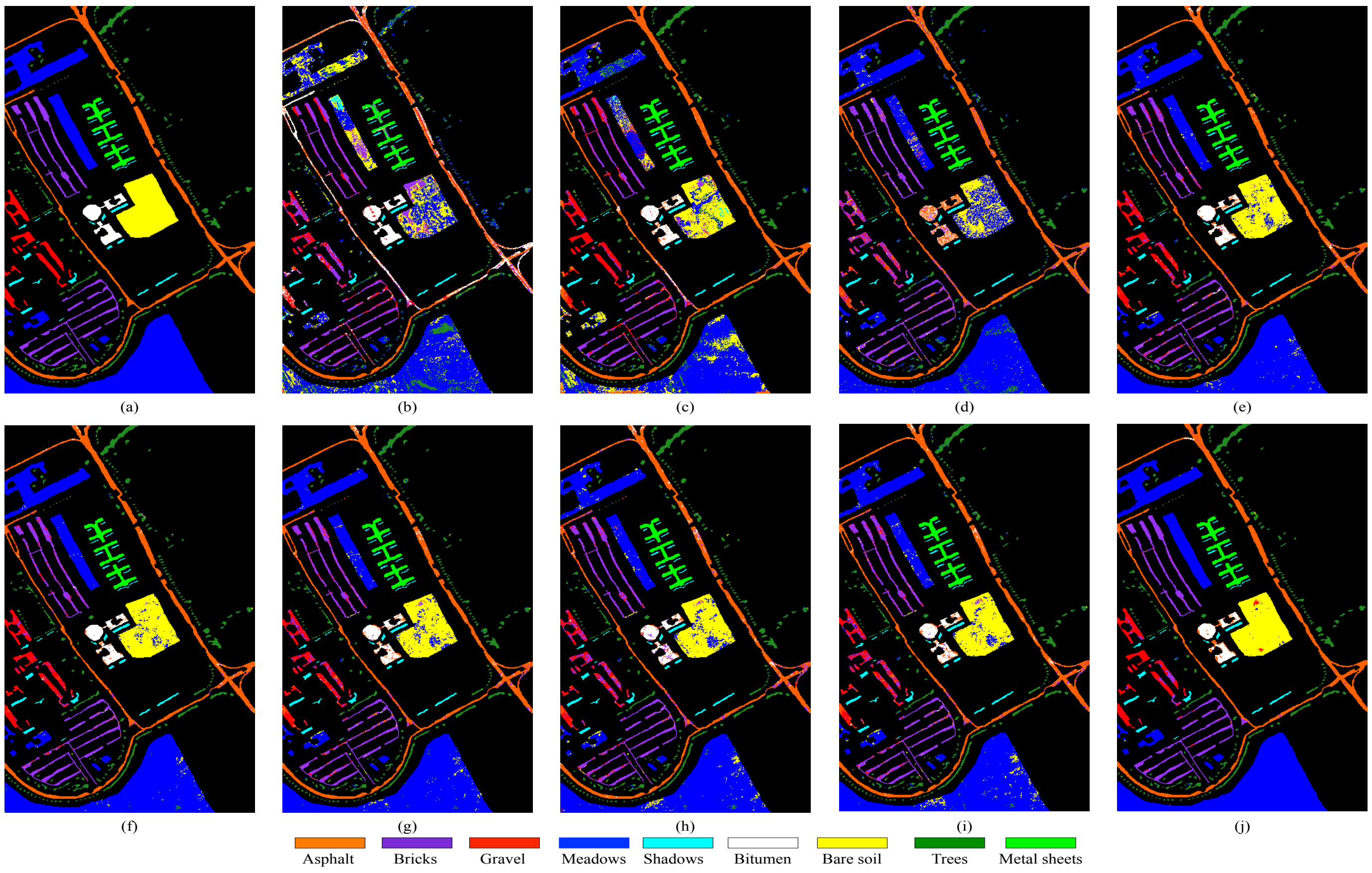

Fig. 10. **Visual classification results for the Pavia University dataset: (a) reference land-cover (b) SC^{3}SVM (c) SS-LPSVM (d) SdAE (e) CNNs (f) 3D-CNNs (g) ppMLR (h) MLRpr (i) ppMLRpr (j) proposed method.**

the benchmark methods, and the procedure for their parameter setting can be found in the corresponding references. Besides, we also provide comparisons with other deep learning methods (including SdAE, CNNs and 3D-CNNs [27]). The spatial size of the input image patches for CNNs and 3D-CNNs was set to 7×7. For the spectral SdAE stream, we used the 3-layer or 4-layer network according to the different datasets, and each hidden layer had 100 units. For the spatial CNNs stream, the kernel size of convolutional layer was 2×2. Because larger kernel size in pooling layer will result in a dramatic image size shrinkage, we set the kernel size of pooling layers to 2×2.

From Tab. V, we can observe that the proposed method achieves significantly higher OA, with gains of 3.29%, 9.04%, 12.61%, 14.54% and 26.23% over CNNs, MLRpr, SdAE, SS-LPSVM, SC^{3}SVM, respectively. The OA of the proposed method is slightly better compared with ppMLRpr, ppMLR and 3D-CNNs. The AA of the proposed method can reach the maximum value of 99.22%, suggesting that the proposed deep architecture can effectively reflect the distribution information of the data. CNNs, 3D-CNNs and ppMLR obtain good AA results, *i.e.*, 96.50%, 97.43% and 98.43%, respectively. Nevertheless, their AA results are lower than those of the proposed method. We can also observe in Tab. V that only 9 and 12 samples were selected as the training sets for the

Grass/pasture-mowed and Oats, respectively. The accuracies of SC^{3}SVM, MLRpr and ppMLRpr for these two classes are greatly influenced by the number of training samples. However, the accuracies of these classes obtained by the proposed method can both achieve 100%. For illustrative purposes, the classification maps are shown in Fig.9. Compared with Fig.9(j), the noise in Fig.9(b)-Fig.9(d) is quite evident. The samples of Soybeans-no till in the upper left corner are misclassified into Soybeans-clean till in Fig.9(g)-Fig.9(i). The class of Oats can best demonstrate the advantages of the proposed method.

The OA, AA and individual class accuracy using the University of Pavia dataset are shown in Tab. VI. The OA and AA of the proposed method are 97.50% and 94.63%, which are significantly higher than those of other classification methods. The accuarcy for each individual stream (*i.e.,* SdAE and CNNs) both have an improvement after the integration by the proposed two-stream architecture. The classification accuracy of 3D-CNNs (94.35% OA) is 3.15% lower than that of the proposed method. The visualization of the classification results are shown in Fig. 10. By analyzing the figure, one can observe that other methods often incur in errors on the Bare soil areas, whereas the proposed two-stream architecture can classify the regions accurately.

TABLE VII
P-VALUES (IN PERCENT) AND WIN/TIE/LOSS COUNTS OF THE PROPOSED METHOD VERSUS OTHER COMPETITIVE METHODS BASED ON OA.

Dataset	3D-CNNs	ppMLR	W/T/L
Indian Pines	0.72/0.36/99.64 (W)	38.25/19.13/80.87 (T)	1/1/0
Pavia University	0.65/0.33/99.67 (W)	0.68/0.34/99.66 (W)	2/0/0
W/T/L	2/0/0	1/1/0	-

We have further implemented the Paired *t*-test at 95% significance level to illustrate that the proposed method is statistically better than the baselines (*i.e.*, 3D-CNNs and ppMLR). The results are shown in Tab. VII. The original hypothesis is that the OA of baseline is compared with the proposed method, where $p = p_1/p_2/p_3$ is composed of p_1 (the OA of the baseline equal to that of the proposed method), p_2 (the OA of the baseline higher than or equal to that of the proposed method) and p_3 (the OA of the baseline lower than or equal to that of the proposed method), respectively. We can see that the proposed method achieves 2 wins, 0 tie and 0 loss when compared to 3D-CNNs, and 1 win, 1 tie and 0 loss when compared to ppMLR. Therefore, from the perspective of statistics, our method has a better overall performance compared with the baselines.

G. Anlysis of the computational efficiency

The analysis of the computational time was carried out on the Pavia University dataset, and the experimental platform was a computer equipped with an Intel Core i5 Processor at 2.70-GHz CPU. We took the test set as the validation set, and the batch size was set to 128. For each epoch, the proposed method takes 12*s* to train the model, in which pre-training and fine-tuning consume 9*s* and 3*s*, respectively. To achieve the best parameters, we need to increase the number of training epochs, which is set to 1000 for the considered dataset. The computational efficiency is higher than the traditional methods as the code is better suited to run on GPU. The GPU-based SVM [58] takes about 127.80*s*, which is 6 times faster than that on a standard CPU-based SVM. However, it is still much slower than the softmax classifier. Therefore, the proposed architecture is computationally efficient in the testing phase, and consumes less time than either the GPU-based SVM or the CPU-based SVM. However, if training can be implemented through the GPU-cuda accelerated Caffe library, the whole training time can be further cut down as shown in [16].

V. CONCLUSION

In this paper, we propose a novel two-stream deep architecture with a class-specific fusion scheme. In the two-stream architecture, the stacked denoising auto-encoder is adopted to encode the spectral features, and deep convolutional neural networks are employed to extract deep spatial features. In the fusion architecture, the prediction probabilities from two streams are aggregated by a class-specific fusion strategy, which learns the class fusion weights adaptively, and takes into account the correlation among different classes. Experimental results demonstrate that the proposed architecture can achieve competitive performance compared with the state-of-art methods.

To sum up, this paper introduces a novel deep learning architecture for the problem of hyperspectral image classification. However, the training of the proposed method which includes the SdAE and CNNs is time-consuming. Therefore, further research can be conducted to explore more efficient computational schemes for the proposed two-stream deep architecture, and explore possible semi-supervised learning techniques.

ACKNOWLEDGMENT

The authors would like to thank Prof. P. Gamba for providng the dataset of Pavia. This research is supported by the Natural Science Foundation of Shandong Province, (No. ZR2017PF004), the National Natural Science Foundation of China (No. 61701272 and No. 61572269) and the Fundamental Research Funds for the Central Universities (No. 2682016CX083).

REFERENCES

[1] G. Camps-Valls, L. Gomez-Chova, J. Munoz-Mari, J. Vila-Frances, and J. Calpe-Maravilla, “Composite kernels for hyperspectral image classification,” *IEEE Geoscience and Remote Sensing Letters*, vol. 3, no. 1, pp. 93–97, Jan 2006.

[2] Y. Chen, Z. Lin, X. Zhao, G. Wang, and Y. Gu, “Deep learning-based classification of hyperspectral data,” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 6, pp. 2094–2107, 2014.

[3] K. Makantasis, K. Karantzalos, A. Doulamis, and N. Doulamis, “Deep supervised learning for hyperspectral data classification through convolutional neural networks,” in *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, 2015, pp. 4959–4962.

[4] G. Camps-Valls, D. Tuia, L. Bruzzone, and J. A. Benediktsson, “Advances in hyperspectral image classification: Earth monitoring with statistical learning methods,” *IEEE Signal Processing Magazine*, vol. 31, no. 1, pp. 45–54, Jan 2014.

[5] A. Plaza, J. A. Benediktsson, J. W. Boardman, J. Brazile, L. Bruzzone, G. Camps-Valls, J. Chanussot, M. Fauvel, P. Gamba, and A. Gualtieri, "Recent advances in techniques for hyperspectral image processing," *Remote Sensing of Environment*, vol. 113, no. 1, pp. S110–S122, 2009.

[6] Y. Tarabalka, J. Chanussot, and J. A. Benediktsson, "Segmentation and classification of hyperspectral images using minimum spanning forest grown from automatically selected markers," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 40, no. 5, pp. 1267–1279, Oct 2010.

[7] G. Moser, S. B. Serpico, and J. A. Benediktsson, "Land-cover mapping by markov modeling of spatial-contextual information in very-high-resolution remote sensing images," *Proceedings of the IEEE*, vol. 101, no. 3, pp. 631–651, March 2013.

[8] J. A. Palmason, J. A. Benediktsson, J. R. Sveinsson, and J. Chanussot, "Classification of hyperspectral data from urban areas using morphological preprocessing and independent component analysis," in *Proceedings. 2005 IEEE International Geoscience and Remote Sensing Symposium, 2005. IGARSS '05.*, vol. 1, July 2005, pp. 4 pp.–.

[9] J. A. Benediktsson, J. A. Palmason, and J. R. Sveinsson, "Classification of hyperspectral data from urban areas based on extended morphological profiles," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 3, pp. 480–491, March 2005.

[10] M. D. Mura, J. A. Benediktsson, B. Waske, and L. Bruzzone, "Extended profiles with morphological attribute filters for the analysis of hyperspectral data," *International Journal of Remote Sensing*, vol. 31, no. 22, pp. 5975–5991, 2010.

[11] M. Shi and G. Healey, "Hyperspectral texture recognition using a multiscale opponent representation," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 41, no. 5, pp. 1090–1095, May 2003.

[12] Y. Yuan, J. Lin, and Q. Wang, "Dual-clustering-based hyperspectral band selection by contextual analysis," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 3, pp. 1431–1445, 2016.

[13] 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, jul 2006.

[14] D. P. Yoshua Bengio, Pascal Lamblin and H. Larochelle, *Greedy Layer-Wise Training of Deep Networks*. MIT Press, 2007, pp. 153–160.

[15] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, Aug 2013.

[16] E. Aptoula, M. C. Ozdemir, and B. Yanikoglu, "Deep learning with attribute profiles for hyperspectral image classification," *IEEE Geoscience and Remote Sensing Letters*, vol. 13, no. 12, pp. 1970–1974, 2016.

[17] X. Ma, H. Wang, and J. Geng, "Spectral-spatial classification of hyperspectral image based on deep auto-encoder," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. PP, no. 99, pp. 1–13, 2016.

[18] X. Han, Y. Zhong, and L. Zhang, "Spatial-spectral classification based on the unsupervised convolutional sparse auto-encoder for hyperspectral remote sensing imagery," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. III-7, pp. 25–31, 2016.

[19] A. Romero, C. Gatta, and G. Camps-Valls, "Unsupervised deep feature extraction for remote sensing image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 3, pp. 1349–1362, March 2016.

[20] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," *CoRR*, vol. abs/1406.2199, 2014.

[21] J. Yang, Y. Zhao, C. W. Chan, and C. Yi, "Hyperspectral image classification using two-channel deep convolutional neural network," in *Geoscience and Remote Sensing Symposium*, 2016, pp. 5079–5082.

[22] E. Othman, Y. Bazi, N. Alajlan, H. Alhichri, and F. Melgani, "Using convolutional features and a sparse autoencoder for land-use scene classification," *International Journal of Remote Sensing*, vol. 37, no. 10, pp. 2149–2167, 2016.

[23] K. T. Lai, D. Liu, S. F. Chang, and M. S. Chen, "Learning sample specific weights for late fusion," *IEEE Transactions on Image Processing*, vol. 24, no. 9, pp. 2772–2783, Sept 2015.

[24] D. Liu, K. T. Lai, G. Ye, M. S. Chen, and S. F. Chang, "Sample-specific late fusion for visual category recognition," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, June 2013, pp. 803–810.

[25] P. K. Atrey, M. A. Hossain, A. E. Saddik, and M. S. Kankanhalli, "Multimodal fusion for multimedia analysis: a survey," *Multimedia Systems*, vol. 16, no. 6, pp. 345–379, 2010.

[26] Y. Chen, C. Li, P. Ghamisi, X. Jia, and Y. Gu, "Deep fusion of remote sensing data for accurate classification," *IEEE Geoscience and Remote Sensing Letters*, vol. PP, no. 99, pp. 1–5, 2017.

[27] Y. Li, H. Zhang, and Q. Shen, "Spectral-spatial classification of hyperspectral imagery with 3d convolutional neural network," *Remote Sensing*, vol. 9, no. 1, p. 67, 2017.

[28] F. Palsson, J. R. Sveinsson, and M. O. Ulfarsson, "Multispectral and hyperspectral image fusion using a 3-d-convolutional neural network," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 5, pp. 639–643, 2017.

[29] W. Li, C. Chen, H. Su, and Q. Du, "Local binary patterns and extreme learning machine for hyperspectral imagery classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 7, pp. 3681–3693, 2015.

[30] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *ICML08*, 2008, pp. 1096–1103.

[31] Y. Ueda, L. Wang, A. Kai, and B. Ren, "Environment-dependent denoising autoencoder for distant-talking speech recognition," *EURASIP Journal on Applied Signal Processing*, p. 92, Dec. 2015.

[32] L. Xugang, T. Yu, M. Shigeki, and H. Chiori, "Ensemble modeling of denoising autoencoder for speech spectrum restoration," in *INTERSPEECH2014*, 2014, pp. 885–889.

[33] S. Araki, T. Hayashi, M. Delcroix, M. Fujimoto, K. Takeda, and T. Nakatani, "Exploring multi-channel features for denoising-autoencoder-based speech enhancement," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2015, pp. 116–120.

[34] A. Budiman, M. I. Fanany, and C. Basaruddin, "Stacked denoising autoencoder for feature representation learning in pose-based action recognition," in *2014 IEEE 3rd Global Conference on Consumer Electronics (GCCE)*, Oct 2014, pp. 684–688.

[35] Z. Lin, Y. Chen, X. Zhao, and G. Wang, "Spectral-spatial classification of hyperspectral image using autoencoders," in *Information, Communications and Signal Processing (ICICS) 2013 9th International Conference on*, 2013, pp. 1–5.

[36] X. Chen, M. Li, and Y. Xiaoquan, "Stacked denoise autoencoder based feature extraction and classification for hyperspectral images," *Journal of Sensors*, p. 10, 2016.

[37] R. Guo, W. Wang, and H. Qi, "Hyperspectral image unmixing using autoencoder cascade," in *Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS), 2015 7th Workshop on*. IEEE, 2015, pp. 1–4.

[38] 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, pp. 3371–3408, Dec. 2010.

[39] C. Kandaswamy, L. M. Silva, L. A. Alexandre, R. Sousa, J. M. Santos, and J. M. de S, "Improving transfer learning accuracy by reusing stacked denoising autoencoders," in *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Oct 2014, pp. 1380–1387.

[40] S. Li, J. Kawale, and Y. Fu, "Deep collaborative filtering via marginalized denoising auto-encoder," in *The ACM International*, 2015, pp. 811–820.

[41] Y. Liu, G. Cao, Q. Sun, and M. Siegel, "Hyperspectral classification via learnt features," in *Image Processing (ICIP), 2015 IEEE International Conference on*, Sept 2015, pp. 2591–2595.

[42] M. Chen, Z. E. Xu, K. Q. Weinberger, and F. Sha, "Marginalized denoising autoencoders for domain adaptation," *Computing Research Repository*, vol. abs/1206.4683, 2012.

[43] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: A deep learning approach," in *ICML*, 2011.

[44] C. Wang, P. Zhang, Y. Zhang, L. Zhang, and W. Wei, "A multi-label hyperspectral image classification method with deep learning features," in *Proceedings of the International Conference on Internet Multimedia Computing and Service*, ser. ICIMCS'16. New York, NY, USA: ACM, 2016, pp. 127–131. [Online]. Available: http://doi.acm.org/10.1145/3007669.3007742

[45] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[46] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014. [Online]. Available: http://arxiv.org/abs/1409.1556

[47] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *CoRR*, vol. abs/1409.4842, 2014. [Online]. Available: http://arxiv.org/abs/1409.4842

[48] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[49] ——, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 9, pp. 1904–1916, September 2015.

[50] D. C. Ciresan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber, “Flexible, high performance convolutional neural networks for image classification,” in *IJCAI 2011, Proceedings of the International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July*, 2011, pp. 1237–1242.

[51] S. Yu, S. Jia, and C. Xu, “Convolutional neural networks for hyperspectral image classification,” *Neurocomputing*, pp. –, 2016.

[52] 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 Transactions on Geoscience and Remote Sensing*, vol. 54, no. 10, pp. 6232–6251, Oct 2016.

[53] H. Liang and Q. Li, “Hyperspectral imagery classification using sparse representations of convolutional neural network features,” *Remote Sensing*, vol. 8, p. 99, jan 2016.

[54] B. C. Kuo, C. S. Huang, C. C. Hung, Y. L. Liu, and I. L. Chen, “Spatial information based support vector machine for hyperspectral image classification,” in *Geoscience and Remote Sensing Symposium*, 2010, pp. 832–835.

[55] W. Liguo, H. Siyuan, W. Qunming, and W. Ying, “Semisupervised classification for hyperspectral imagery based on spatialspectral label propagation,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 97, pp. 123–137, 2014.

[56] 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 Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 9, no. 2, pp. 625–639, Feb 2016.

[57] J. Li, J. Bioucas-Dias, and A. Plaza, “Hyperspectral image segmentation using a new bayesian approach with active learning,” *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 49, no. 10, pp. 3947–3960, oct. 2011.

[58] A. Athanasopoulos, A. Dimou, V. Mezaris, and I. Kompatsiaris, “Gpu acceleration for support vector machines,” in *WIAMIS 2011: 12th International Workshop on Image Analysis for Multimedia Interactive Services, Delft, The Netherlands, April 13-15, 2011*, 2011.