Convolutional neural network extreme learning machine for Effective Classification of hyperspectral images

Faxian Cao[1], Zhijing Yang[1*], Jinchang Ren[2], Wing-Kuen Ling[1]
[1] School of Information Engineering, Guangdong University of Technology, Guangzhou, 510006, China; faxiancao@foxmail.com; yzhj@gdut.edu.cn; yongquanling@gdut.edu.cn
[2] Department of Electronic and Electrical Engineering, University of Strathclyde, Glasgow, G1 1XW, UK; jinchang.ren@strath.ac.uk
* Correspondence: yzhj@gdut.edu.cn; Tel.: +86-20-39322438

Abstract—Due to its excellent performance in terms of fast implementation, strong generalization capability and straightforward solution, extreme learning machine (ELM) has attracted increasingly attentions in pattern recognition such as face recognition and hyperspectral image (HSI) classification. However, the performance of ELM for HSI classification remains a challenging problem especially in effective extraction of the featured information from the massive volume of data. To this end, we propose in this paper a new method to combine convolutional neural network (CNN) with ELM (CNN-ELM) for HSI classification. As CNN has been successfully applied for feature extraction in different applications, the combined CNN-ELM approach aims to take advantages of these two techniques for improved classification of HSI. By preserving the spatial features whilst reconstructing the spectral features of HSI, the proposed CNN-ELM method can significantly improve the accuracy of HSI classification without increasing the computational complexity. Comprehensive experiments using three publicly available HSI datasets, Pavia University, Pavia center, and Salinas have fully validated the improved performance of the proposed method when benchmarking with several state-of-the-art approaches.

Keywords—Hyperspectral image (HSI) classification, Convolutional neural network (CNN), Extreme learning machine (ELM), Pattern recognition.

1 Introduction

With spectral information in hundreds of continuous narrow bands and spatial information acquired simultaneously, hyperspectral imaging has facilitated a number of applications, such as agricultural, military defense, agriculture, medical diagnosis and analyses of crime scene details (Khan et al; 2018, Boldrini et al, 2012), especially in remote sensing earth observation. As the spectral profiles can reflect certain physical (i.e. moisture/temperature) or chemical differences of the objects, this has been widely used in land mapping for classification of the images (Khan et al, 2018). Although HSI data classification is conceptually similar to image labeling in computer vision (Wang et al. 2015), one fundamental challenge here is the curse of dimensionality or Hughes phenomenon (Hughes. 1968) caused by limited labeled data samples (in spatial domain) but too many spectral bands (feature dimensions) (Yu et al. 2017; Sun et al. 2016).

To tackle this problem, a number of techniques have been proposed for feature extraction and dimensionality reduction (Yuan et al. 2017; Du et al. 2016, such as principal component analysis (PCA) (Zabalza et al. 2014), singular spectrum analysis (SSA) (Qiao et al. 2017; Zabalza et al. 2015; Qiao et al. 2015), Low-Rank Representation (Lu et al. 2013), and segmented auto-encoder (Zabalza et al. 2016). For data classification, typical approaches include support vector machine (SVM) (Melgani and Bruzzone 2004), multi-kernel classification (Fang et al. 2015), k-nearest-neighbors (k-NN) (Ma et al. 2010) and multinomial logistic regression (MLR) (Li et al. 2012, 2010) et al. Among these approaches, spatial-spectral analysis becomes a trend as it extracts information in both spatial domain and spectral domain. Whilst spectral information measures the physical/chemical characteristics, it is the spatial structuring information that groups pixels into objects. Therefore, fusion of these two modalities of information is essential for classification of HSI data.

For effective spatial-spectral analysis of HSI, convolutional neural network (CNN) based deep learning is employed for its

success in feature extraction and extraction of the hidden structures of the data (Hinto and Salakhutdinov 2006). As one of the most popularly used model in deep learning, CNN can exploit spatially local correlation by enforcing a local connectivity pattern between neurons of adjacent layers (Hu et al. 2015; Fukushima 1988; Lecun et al. 1998; Ciresan et al. 2011; Simard et al. 2003). Although CNN has already been successfully applied for HSI classification (Slavkovikj et al. 2015; Yue et al. 2015; Makantasis 2015), the training process is over complicated due to the lengthy iterations over the high data volume. For practical applications especially with airborne or satellite based systems, the computational cost needs be cut down to the meet the requirement for real-time data analysis.

In this paper, a convolutional neural network extreme learning machine (CNN-ELM) approach is proposed for hyperspectral image classification. Inspired by the reference (Khan et al. 2017), we get the idea of the complete architecture and the purpose of each layer and training parameters of CNN. Rather using a lengthy process for iterative feature extraction, we apply CNN for only one iteration in training, followed by ELM for data classification under significantly reduced time for feature extraction. CNN has been combined with other methods for classification. For example, CNN and SVM has been combined and successfully applied for HSI classification, where CNN was used for feature extraction followed by SVM for classification (Leng et al 2016). However, compared with SVM, ELM has more efficient computation and comparable classification accuracy (Li et al, 2015). As a single-hidden layer feedforward neural network, ELM has been successfully applied in a number of application areas for merits in terms of fast implementation, straightforward solution and strong generalization capability (Huang et al. 2004; Bai et al. 2014; Huang et al. 2012). As a result, the combination of these two methods is expected to produce much improved data classification results in our proposed CNN-ELM approach. Furthermore, as can be seen from the references (Khan et al, 2018; Boldrini et al, 2012), HSI classification methods can be applied for many applications. Hence, the proposed methods can also be applied for many applications, such as land map classification, medical diagnosis, etc.

The main contributions of the proposed CNN-ELM approach can be highlighted as follows. First, the combination itself is rare, especially for HSI classification with CNN used for feature extraction and ELM for data classification. Second, the proposed method can not only reconstructs the spectral features but also preserve the spatial information. Third, applying CNN only for one iteration has significantly reduced the computational cost whilst still improved the classification accuracy. The experiment results on three well-known publicly available HSI datasets, Pavia University, Pavia center, and Salinas, have validated the efficacy of the proposed approach when benchmarking with several the-state-of-art techniques.

The remainder of this paper is organized as follows. Section 2 introduces briefly the background knowledge of ELM and CNN. Section 3 discusses in detail the proposed CNN-ELM approach in three steps, i.e. normalization, CNN based spectral feature reconstruction and ELM based classification. Experimental results and analysis are presented in Section 4, followed by some concluding remarks drawn in Section 5.

2 Reviews of ELM and CNN for Data Classification in HSI

In this section, the background knowledge of CNN and ELM is presented. Discussions are followed to show how they can be applied in HSIs for data classification.

2.1 Background introduction of ELM

Let $\mathbf{x} = (\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_\mathrm{N}) \in \mathrm{R}^{\mathrm{N}\times\mathrm{d}}$ denote training samples of a HSI, which has N samples of spatial pixels and each sample is a d-dimension vector, we also define $\mathbf{y} = (\mathrm{y}_1, \mathrm{y}_2, \ldots, \mathrm{y}_\mathrm{N}) \in \mathrm{R}^{\mathrm{N}\times\mathrm{M}}$ as the desired output of M different labels for the N samples. As shown in Fig. 1, ELM is a single-hidden layer feedforward neural network, and an ELM with L hidden nodes can be modeled as (Huang et al. 2011):

$$\sum_{j=1}^{L} \beta_j G(we_j^T x_i + bi_j) = y_i \tag{1}$$

where T is the transpose operation, $we_i = (we_{i1}, we_{i2}, \ldots, we_{id})$ and $\mathrm{bi_i}$ are respectively the weight vector and the bias

connecting the input layer and hidden layer of the *i*-th sample of his. In addition, β_j is the ouput weight vector of *i*-th sample of his, and g is the activation function.

For data classification, there are three key steps in ELM as detailed below.

Step1: Assign random inputs for the weight vector we_i and the bias bi_i, where $i = 1, 2, \ldots, L$.

Step2: Using (1) to calculate the output matrix of the hidden layer G

where

$$G(we_1, we_2, \ldots, we_L; x_1, x_2, \ldots, x_N; bi_1, bi_2, \ldots, bi_L) = \begin{bmatrix} g_{11}({we_1}^T x_1 + b_1) & \ldots & g_{1L}({we_L}^T x_1 + bi_L) \\ \ldots & \ldots & \ldots \\ g_{N1}({we_1}^T x_N + b_1) & \ldots & g_{NL}({we_L}^T x_N + bi_L) \end{bmatrix} \quad (2)$$

Step 3: Calculate the output weight matrix $\beta = [\beta_1, \ldots, \beta_L]_{L \times M}$ by

$$\beta = G^{\dagger} y. \quad (3)$$

where $G^{\dagger}$ denotes Moore-Penrose generalized inverse of matrix *G* (Huang et al. 2004), and *y* represents the desired output in (1).

Any piecewise continual function can be used as the hidden layer activation function (Huang et al. 2004). The input weight and bias of ELM are randomly generated and The output weight matrix can be computed as $\beta = G^{\dagger} * y$, so the time-consuming can be greatly reduce.

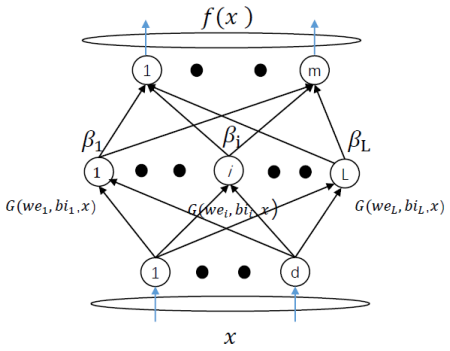


Fig. 1. The architecture of an ELM.

2.2 Background introduction of CNN

CNN is considered to be one of the relatively successful machine learning methods because of its good performance. As shown in Fig. 2, a typical CNN consists of several layers (Hu et al. 2015; Sainath et al. 2013). The first layer is the input layer, while the second and third layers are the convolution layer and the max pooling layer, respectively. The convolution layer convolutes the input data to form the feature map to reduce the training parameters. That is to say, each hidden activation function of CNN is computed by multiplying a small local input with the weights *W*. The neurons belonging to the same layer share the same weights, which can be describe as follows:

$$h_i = W * (x_i + x_{i+1} + x_{i+2}) + b_i \quad (4)$$

where b_i is the bias of the convolutional layer. The max pooling layer partitions the feature map from convolutional layer into a set of non-overlapping windows and outputs the maximum value. The final layer is a fully connected layer which outputs the classification results.

2.3 Adapting CNN and ELM in HSI

Comparing with SVM and other state-of-the-art data classification algorithms, ELM is considered as a promising method with the following advantages (Samat et al. 2014). Firstly, it has a simpler structure and higher generalization performance than SVM and most others. Secondly, it has a very high computational efficiency for greatly shortened training time. Thirdly, it needs no tuning of additional parameters when the network structure is set. Fourthly, there are many available piecewise continual functions which can be used as the activation function, such as sine function, radial basis function and sigmoid function, etc. As a result, ELM has been successfully applied in many applications (Samat et al. 2014). However, the classification results are not high when applying ELM directly to HSI. The reason of low recognition rate mainly is that the ELM cannot catch the depth features of HSI. For example, as reported in (Lv et al. 2016), the overall classification accuracy of ELM for Pavia University datasets is only 79.58%. Therefore it is a critical problem how to maintain fast speed of ELM and improve the accuracy for HSI classification.

As mentioned above, CNN can extract the spectral features of depth of HSI data very well. So we use CNN to extract the depth feature of HSI, then the reconstructed pixels of HSI are used as the input of ELM. The combination of these two methods is expected to obtain good classification results and maintain the fast speed for HIS classification.

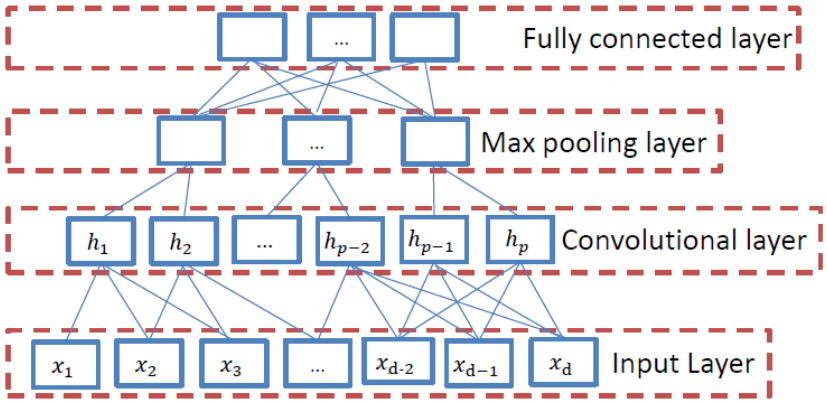


Fig. 2. A typical architecture of CNN consists of input layer, convolutional layer, max pooling layer and fully connected layer.

3 The Proposed CNN-ELM Approach

The proposed method can be divided into three parts: normalization, spectral feature reconstruction using CNN, and classification using ELM.

3.1 Normalization

Let $x \equiv (x_1, x_2, \ldots, x_N) \in R^{N \times L}$ be a HSI data that has N samples and L feature. Normalization is a preprocessing process that it makes the HSI data remain in the range of [0, 1]. Since the normalization is an important preprocessing step for HSI classification, many normalization approaches have been proposed. Here we choose the following widely used normalization method (Li et. al. 2015), which can be described as follows:

$$x_{ij} = \frac{x_{ij}}{\max(X)} \quad \mathrm{i} = 1, 2, \ldots, \mathrm{N}; \mathrm{j} = 1, 2, \ldots, \mathrm{L} \tag{5}$$

where x_{ij} is any pixel of the HSI data, and *max()* gets the largest one of all the data.

3.2 Spectral feature reconstruction using CNN

In order to maintain the high speed of the algorithm, we let CNN iterate only one time to reduce the consuming time. The hierarchical structure of CNN has been shown to be the most successful and efficient method to learn visual features. HSI data

have hundreds of spectral bands so that we can think of the spectral feature of pixels as a two-dimensional curve. We use CNN to extract the spectral feature of the depth of the pixel to reconstruct the spectral feature, and then improve the classification accuracy of ELM with little consuming time.

Table 1 The architecture of CNN we used.

Layer	Type	Number of maps and neurons and maps	Kernel Size	Stride
1	Input	1 map of n1 neurons		
2	Convolutional	10 map of n2 neurons	K1	1
3	Max pooling	10 map of n3 neurons	K2	1
4	Convolutional	20 map of n4 neurons	K3	1
5	Max pooling	20 map of n5 neurons	K4	1
6	Rasterization	n6 neurons		
7	Fully-connected	n7 neurons		
8	Output	n8 neurons		

As show in Table 1, CNN consists of eight layers. The first layer is the input layer which represents the spectral vector of one pixel of HSI dataset. The second and third layers are the convolution layer and the max pooling layer, respectively. The fourth layer and the fifth layer are also convolution and max pooling layer. The data from convolutional layer after max pooling operation is a series of feature map, but the input received by the multi-layer perceptron is a vector. So the elements in these feature maps should be arranged in a vector. The sixth layer is the rasterization layer which is a fully connected layer, followed by another fully connected layer, and then the final layer is the output layer. Table 1 shows an example of the number of maps/filters and the kernel sizes in each layer, where K1, K2, K3 and K4 are the kernel size in the corresponding layer, which will be given the exact sizes for different datasets in the experimental section. The output layer of CNN is just used for training. The purpose is to update the weights and bias for back propagation, which would allow deeper features to be extracted. We will not use the output layer when we test our labeled sample. θ is assumed to represent all training parameters, θ={θ_i} and i=2,3,4,5,6,7,8 where θ_i is the parameters set between the *(i-1)*-th and the *i*-th layers.

Assuming x_i is the input of the i-th layer and the output of the th (i+1)-th layer, we can compute x_{i+1} by the following formula:

$$x_{i+1} = f_i(u_i) \tag{6}$$

where

$$u_i = w_i^T x_i + b_i \tag{7}$$

and T is transpose operation, w_i and $b_i$is the weight matrix and bias of the ith layer acting on the input data, respectively.

For the output layer, we use softmax function as the activation function, which is defined as:

$$\mathrm{y} = \frac{1}{\sum_{k=1}^{n7} e^{w_{L,k}^T x_L + b_{L,K}}} \begin{bmatrix} w_{L,1}^T x_L + b_{L,1} \\ \dots \\ w_{L,n7}^T x_L + b_{L,n7} \end{bmatrix}. \tag{8}$$

The back propagation updates the weights until the error is acceptable. The error is the deviation between the actual response of the training sample in the forward propagation phase and the target output corresponding to the sample. The training parameters are updated by minimizing the loss function which is achieved by gradient descent. The loss function in our work is defined as follows:

$$J(\theta) = -\frac{1}{P}\sum_{i=1}^{p}\sum_{j=1}^{n7} 1\{j = Y^{(i)}\}\log(y_j^{(i)}), \tag{9}$$

where p is the total number of training samples, Y and $y_j^{(i)}$ are the desired output and the actual output of the *j*-th sample, respectively. The probability value of the desired output of the *j*-th sample is 1, and the probability values of the others are 0. The expression $1\{j = Y^{(i)}\} = 1$ if *j* is equal to the desired output $Y^{(i)}$ of the *i*-th training sample, otherwise its value is equal to 0. The training parameters are update by the following equation:

$$\theta = \theta - \alpha\nabla_\theta J(\theta) \tag{10}$$

where α is the learning factor which is set to be 0.05 in our experiment, and

$$\nabla_\theta J(\theta) = \left\{\frac{\partial J}{\partial\theta_1}, \frac{\partial J}{\partial\theta_2}, \ldots, \frac{\partial J}{\partial\theta_L}\right\} \tag{11}$$

and

$$\frac{\partial J}{\partial\theta_i} = \left\{\frac{\partial J}{\partial w_i}, \frac{\partial J}{\partial b_i}\right\}. \tag{12}$$

3.3 Classification using ELM

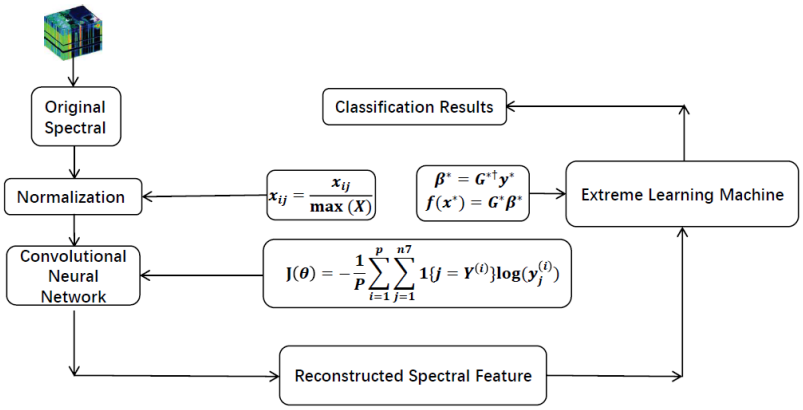


Fig.3 The flowchart of the proposed CNN-ELM.

As mentioned above, when applying to HSI dataset, ELM can't extract the spectral feature of depth. This causes low recognition rate. To improve the accuracy, we use CNN to reconstruct the spectral features. Then the spectral features of depth are used as the input of ELM. Let $x^* \equiv (x_1, x_2, \ldots, x_N) \in R^{N\times Q}$ be the reconstructed spectral feature datasets, i.e., every pixel of HSI is reconstructed to be Q-dimensions, $y = (y_1, y_2, \ldots, y_N)R^{N\times M}$ be the corresponding target label, L be the hidden neuron numbers and $g^*(we^*x^* + bi^*)$ be the activation function of hidden layer, then the process of classification by ELM can be described as follows:

Step1: Generate the input weight matrix $w^*$and bias vector b^* randomly using the uniform distribution function.

Step2: Compute the output matrix of the hidden layer,

$$G^*(we_1^*, we_2^*, \ldots, we_L^*; x_1^*, x_2^*, \ldots, x_N^*; bi_1^*, bi_2^*, \ldots, bi_L^*)$$
$$= \begin{bmatrix} g_{11}(we_1^{*T}x_1^* + bi_1^*) & \ldots & g_{1L}(we_L^{*T}x_1^* + bi_L^*) \\ \ldots & \ldots & \ldots \\ g_{N1}(we_1^{*T}x_N^* + bi_1^*) & \ldots & g_{NL}(we_L^{*T}x_N^* + bi_L^*) \end{bmatrix}. \tag{13}$$

Step3: Calculate the output weights

$$\beta^* = G^{*\dagger}y^* \tag{14}$$

where $$\beta^* = \begin{bmatrix} \beta_1^* \\ \dots \\ \beta_L^* \end{bmatrix}_{L\times M} \tag{15}$$

and † is the Moore-Penrose generalized by the inverse of the hidden layer matrix .

The final classification result can be expressed by the following equation:

$$f(x^*) = G^*\beta^*. \tag{16}$$

We use different numbers of hidden nodes of ELM for different HSI datasets. Better results are achieved by using different hidden nodes according to different HSI data. Fig. 3 shows the flow chart of our proposed method.

4 Experiments and Analysis

In this section, we apply the proposed method to three well known HSI datasets. We use different architectures of CNN for different HSI datasets. The CNN architectures of Pavia University, the CNN architectures of Pavia Center, and the architectures of Salinas are shown in Table 2. The architectures of CNN we used are very effective and our experiment results in three well known HSI datasets demonstrate the feasibility of the architecture.

Table 2 The architecture of CNN with Pavia University, Pavia Center, Salinas

		CNN architecture parameters of Pavia University			CNN architecture parameters of Pavia Center			CNN architecture parameters of Salinas		
Layer	Type	Numbers of maps and neurons	Kernel size	stride	Numbers of maps and neurons	Kernel size	stride	Numbers of maps and neurons	Kernel size	stride
1	Input	1 map of 103 neurons			1 map of n1 neurons			1 map of 204 neurons		
2	Convolutional	10 map of 98 neurons	6×1	1	10 map of 102 neurons	5×1	1	10 map of 196 neurons	9×1	1
3	Max pooling	10 map of 49 neurons	2×1	1	10 map of 49 neurons	2×1	1	10 map of 98 neurons	2×1	1
4	Convolutional	20 map of 44 neurons	6×1	1	20 map of 44 neurons	6×1	1	20 map of 90 neurons	9×1	1
5	Max pooling	20 map of 22 neurons	2×1	1	20 map of 22 neurons	2×1	1	20 map of 45 neurons	2×1	1
6	Rasterization	440			440 neurons			900 neurons		
7	Fully-connected	100			100 neurons			204 neurons		
8	Output	9			9 neurons			16 neurons		

4.1 Dataset

The three HSI datasets and the experiment results are described as follows:

(1) ROSIS Pavia University HSI:

The first *HSI (Li et al. 2013) dataset was collected in 2001 by the Reflective Optics System Imaging Spectrometer (ROSIS) optical sensor which provides 103 bands after removing 12 noisiest bands with a spectral range coverage ranging*

from 0.43 to 0.86 um. The size of the image in pixels is 610×340 with very high spatial resolution of 1.3 m and 9 ground truth classes. The numbers of training samples is 3921 (about 9%) of all labeled data, and all the labeled data are used for testing. Table 3 shows the number of train samples and test samples in our experiments.

(2) ROSIS Pavia Center HSI:

The second HSI (Sun et al. 2015) dataset was the other urban image collected in 2001 by the ROSIS sensors over the center of the Pavia city. The dataset has 1096×715 pixels which each has 102 spectral bands after removing 13 noisy bands. There are also nine classes of images, and the number of training and test samples of each class of the HSI in our experiments is shown in Table 3. There are about 7456 labeled samples used for training, which accounts for about 5 percent of the total sample. In order to compare the classification accuracy with other state-of-the-art methods, we use the rest of the labeled samples for testing.

(3) AVIRIS Salinas HSI:

The third HSI (Zhang et al. 2016) dataset was collected by the AVIRIS sensor over Salinas Valley, California. The image has 214 pixels and every pixel has 224 bands. After removing 20 water absorption bands of spectral, only 204 bands in each pixel. There are 16 classes in the ground truth image and the number of training and test is shown in Table 3. In order to compare with other state-of-the-art method, we also use rest labeled samples for testing.

Table 3 The number of training sample and test sample of Pavia University, Pavia Center and Salinas.

Pavia University			Pavia Center			Salinas					
Class	Train	Test	Class	Train	Test	Class	Train	Test	Class	Train	Test
Asphalt	548	6631	Water	824	65147	Brocoli_green_weed_1	200	1809	Soil_vinyard_develop	620	5583
Meadows	540	18649	Trees	820	6778	Brocoli_green_weed_2	372	3354	Corn_sensced_green_weeds	327	2951
Gravel	392	2099	Meadows	824	2266	Fallow	197	1779	Lettuce_romaine_4wk	106	962
Trees	524	3064	Bricks	808	1891	Fallow_rough_plow	139	1255	Lettuce_romaine_5wk	192	1735
Metal sheets	265	1345	Soil	820	5764	Fallow_smooth	267	2411	Lettuce_romaine_6wk	91	825
Bare soil	532	5029	Asphalt	816	8432	Stubble	395	3564	Lettuce_romaine_7wk	107	963
Bitumen	375	1330	Bitumen	808	6479	Celery	357	3222	Vinyard_untrained	726	6542
Bricks	514	3682	Tiles	1260	41566	Grapes_untrained	1127	10144	Vinyard_vertical_treils	180	1627
Shadows	231	947	Shadows	476	2387						

*It is worth noting that in our experiments, the final output layer of the CNN architectures is only used during training. It facilitates the update of the weights and bias in the back pro*pagation process, so that it can extract spectral feature of depth. We do not need to use the final output layer in the test process. We directly use the reconstructed spectral feature of the seventh layer as input of ELM. In order to maintain the high speed of the algorithm, we let CNN iteration for only one time to reduce the consuming time in the experiment. It is found that it can obtain high classification accuracy with little consuming time. For the three HSI datasets, all the training samples are randomly selected, and all the experiment results were averaged by ten times in Monte Carlo runs. The number of hidden neuron of ELM in each HSI dataset is shown in Table 4.

4.2 The experiments results and analysis of Pavia University dataset

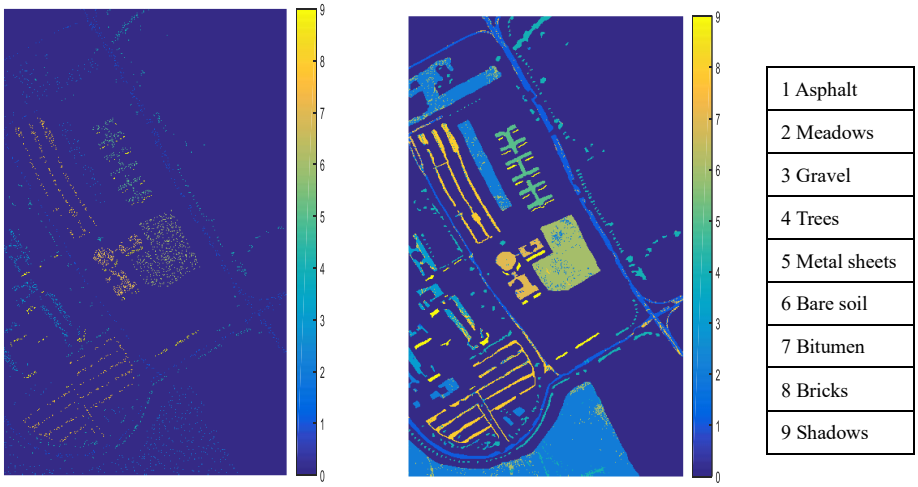
In this subsection, we evaluate the proposed method by comparing with other state-of-the-art methods of HSI classification

using the University of Pavia dataset. Fig. 4 (a) and (b) show the training sample and the classification results with 3921 training samples and all the labeled samples, respectively. Table 5 shows the OA (overall accuracy), AA (average accuracy), k (kappa coefficient) and individual class accuracies of the proposed method and other state-of-the-art methods. In contrast to other methods, our proposed method obtains the best results with the same training samples (about 9% of available samples). Table 3 shows the number of training samples and test samples of Pavia University dataset in this experiment.

Table 4 The hidden nodes of ELM after CNN reconstruct pixel of HSI.

HSI dataset	The numbers of hidden nodes
Pavia University	900
Pavia Center	900
Salinas	1100

Compared with ELM (Lv et al. 2016), our proposed method is superior to ELM for the classification accuracy of each class. In Table 5, we can see that for each class, we improve all the classification accuracy, and for the OA, AA, k, we improve 13.72%, 9.85%, 17.95%, respectively. It shows that our method improves the classification accuracy a lot.



(a) (b)

Fig. 4. Pavia University dataset: (a) Training samples; (b) Testing classification results.

Table 5 PAVIA University: Overall, Average, and individual class accuracy (in percent) and k statistic of different classification methods with 9% training samples. The best accuracy in each row is show bold.

Class	SVM	EMP	LORSAL	Watershed[#]	LORSAL	MPM	*SMLR*	ELM[Δ]	CNN-ELM
	− CK[†]	/SVM[‡]	− CK[†]		− MLL[†]	− LBP[⊗]	− *SpATV*[†]		
Asphalt	79.85	95.36	77.17	93.64	88.48	**95.70**	94.57	77.27	89.54
Meadows	84.68	63.72	81.61	**97.35**	76.22	73.27	82.56	77.53	94.14
Gravel	81.87	**98.87**	82.42	96.23	73.56	74.18	81.13	80.14	86.51
Trees	96.36	95.41	95.46	97.92	**98.76**	97.85	95.01	95.69	97.17
Metal sheets	99.37	87.61	99.03	66.12	99.70	99.85	**100.0**	99.69	98.94
Bare soil	93.55	80.33	96.94	75.09	97.47	98.55	**100.0**	80.47	94.02
Bitumen	90.21	99.48	93.83	**99.91**	94.74	97.97	99.17	82.97	95.17

Bricks	92.81	97.68	94.65	96.98	96.66	**98.89**	98.45	70.27	91.16
shadows	95.35	98.37	97.47	98.56	99.37	93.56	95.45	93.49	**99.49**
OA	87.18	85.22	86.16	85.42	85.69	85.78	90.01	79.58	**93.30**
AA	90.47	90.76	90.95	91.31	91.66	92.20	**94.04**	84.17	94.02
k	83.3	80.86	82.40	81.30	81.90	82.05	87.2	73.26	**91.21**

Notes:

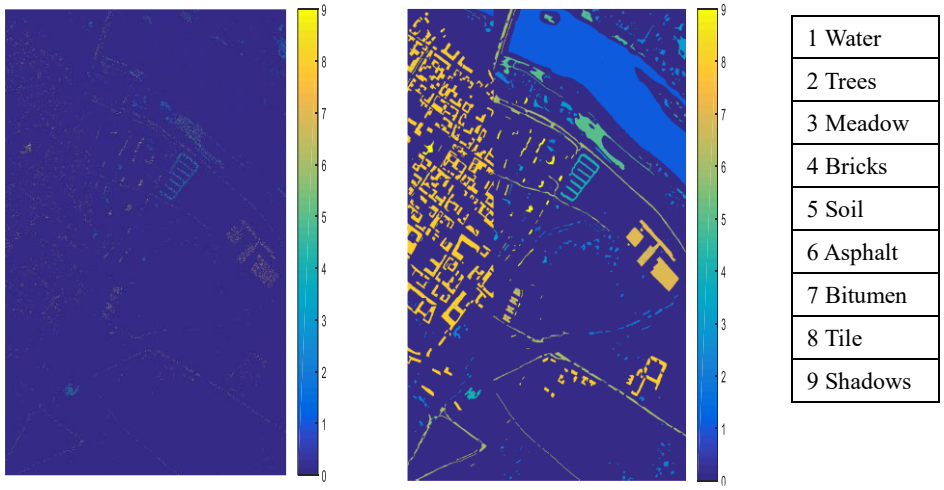
The results of $\mathrm{SVM}-\mathrm{CK}^{\dagger}$ (SVM combined with CK (composite kernel)), $\mathrm{LORSAL}-\mathrm{CK}^{\dagger}$, $\mathrm{LORSAL}-\mathrm{MLL}^{\dagger}$(LORSAL combined with multilevel logistic spatial prior) and $\mathrm{SMLR}-\mathrm{SpATV}^{\dagger}$ (sparse multinomial logistic regression combined with Markov random field) are directly taken from (Sun et al. 2015).

The results of $\mathrm{EMP/SVM}^{\ddagger}$ (EMPs combined with SVM) are directly taken from (Plaza et al. 2009).

The results of $\mathrm{Watershed}^{\#}$ are directly taken from (Tarabalka et al. 2010).

The results of $\mathrm{MPM}-\mathrm{LBP}^{\otimes}$ are directly taken from (Li et al. 2013). The results of ELM^{Δ} are taken from (Lv et al. 2016).

CNN-ELM is the proposed method.



(a) (b)

Fig. 5. Pavia Center dataset: (a) Training samples; (b) Testing classification results.

4.3 The experiments results and analysis of Pavia Center dataset

In this experiment of HSI datasets, we evaluate the classification accuracy of the proposed method by comparing with other state-of-the-art methods of HSI classification. Fig. 5 (a) and (b) show the training sample and the classification results of the proposed method with 7456 training samples and remaining samples, respectively. Table 6 shows the OA (overall accuracy), AA (average accuracy), k (kappa coefficient), and each class' accuracy. In contrast to other methods, the experiment results demonstrate our proposed method yields the best results with the same training samples (about 5% of available samples) and test samples. The number of training samples and test samples of this experiment is shown in Table 3. The experiment results demonstrate our proposed method achieves higher accuracies than other method.

Compared with ELM (Li et al. 2017) in the Table 6, we can see that our proposed method not only improve the classification accuracies of each class, but also improve the OA, AA, and k. For the OA, AA, and k, we improve 4.33%, 12.98% and 8.23%, respectively. The experiment results demonstrate the feasibility of the proposed method again.

Table 6 PAVIA Center: Overall, Average, and individual class accuracy (in percent) and k statistic of different classification methods with 5%

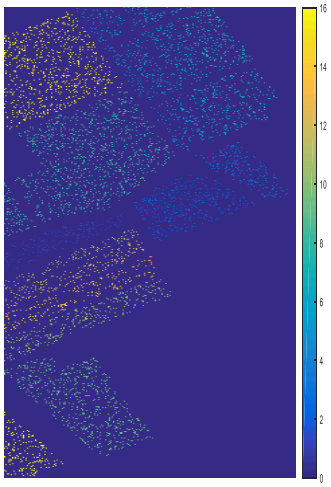
training samples. The best accuracy in each row is shown in bold.

Class	DAFE†	DBFE†	OMP‡	SOMP‡	FOMP‡	ELM‡	**CNN-ELM**
Water	98.9	96.9	99.21	**99.87**	99.97	98.54	99.86
Trees	88.3	91.2	87.70	87.93	87.70	88.35	**95.45**
Meadow	96.3	95.9	95.92	**97.68**	97.15	92.31	96.93
Bricks	**99.6**	98.8	81.27	73.60	83.38	76.31	97.18
Soil	**98.5**	98.4	94.08	96.67	95.51	89.51	96.52
Asphalt	**99.2**	98.6	80.15	77.44	78.66	94.09	97.88
Bitumen	**99.4**	99.1	91.09	94.75	92.98	84.32	94.68
Tile	**99.7**	99.7	97.79	98.48	98.62	95.27	99.12
shadows	63.6	**100**	74.72	83.20	95.53	46.85	99.90
OA	98.05	97.83	95.45	96.20	96.56	94.52	**98.85**
AA	93.71	**97.66**	89.10	89.96	92.17	84.52	97.50
k	97.17	96.88	91.74	93.07	93.73	90.11	**98.34**

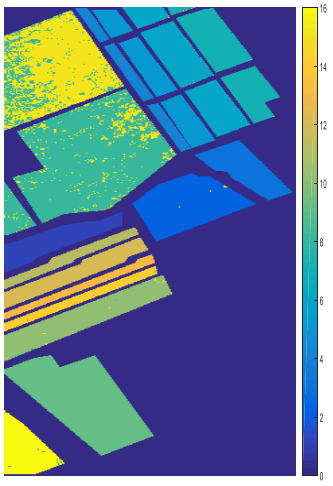
Notes:

The results of DAFE† (Use the mean vector and the covariance matrix of each class for classification) and DBFE†(Features are extracted from the decision boundary between two classes) are taken from (Ghamisi et al. 2014).

The results of OMP‡ (Orthogonal Matching Pursuit), SOMP‡ (Simultaneous Orthogonal Matching Pursuit), FOMP‡ (First-order neighborhood system weighted constraint OMP), ELM‡ are directly taken from (Li et al. 2017).



(a)



(b)

1 Brocoli_green_weed_1	9 Soil_vinyard_develop
2 Brocoli_green_weed_2	10 Corn_sensced_green_weeds
3 Fallow	11 Lettuce_romaine_4wk
4 Fallow_rough_plow	12 Lettuce_romaine_5wk
5 Fallow_smooth	13 Lettuce_romaine_6wk
6 Stubble	14 Lettuce_romaine_7wk
7 Celery	15 Vinyard_untrained
8 Grapes_untrained	16 Vinyard_vertical_treils

Fig. 6 Salinas dataset: (a) Training samples; (b) Testing classification results.

4.4 The experiments results and analysis of AVIRIS Salinas dataset

In this HSI set of experiment, we evaluate our proposed method using the Salinas datasets. Table 7 shows the OA, AA and k statistic of our methods and the other methods using 10% training samples. Fig. 6 (a) and (b) show the training sample and the classification results of the proposed method with 5403 training samples and remaining samples, respectively. Table 3 shows the number of training samples and the test samples of each class. It can be seen that our proposed method achieved better performance than other state-of-the-art HSI classification method. From Table 7, we can see that the proposed method achieves better performance than ELM (Li et al. 2017). For the OA, AA, k, the proposed method is higher than ELM with 6.22%, 4.98%, 6.94%, respectively.

Table 7. SALINAS: overall, average, and individual class accuracy (in percent) and k statistic of different classification methods with 10% training samples. The best accuracy in each row is shown in bold.

Class	SR†	KSR†	SVM‡	OMP‡	SOMP‡	ELM‡	CNN-ELM
Brocoli_green_weed_1	99.72	99.61	99.5	99.50	**99.78**	99.61	99.83
Brocoli_green_weed_2	99.34	99.28	**100**	99.43	99.52	97.17	99.70
Fallow	97.58	97.47	98.99	96.68	97.81	91.57	**99.78**
Fallow_rough_plow	99.52	99.52	99.44	99.60	99.36	90.52	**99.68**
Fallow_smooth	98.26	98.18	**99.17**	97.06	96.43	93.28	98.80
Stubble	99.75	99.69	**99.94**	99.89	99.86	99.55	99.52
Celery	**99.84**	99.78	99.72	99.60	99.41	98.98	99.44
Grapes_untrained	87.67	89.73	**89.79**	78.77	82.29	81.66	89.51
Soil_vinyard_develop	99.73	99.70	99.80	99.12	99.44	96.96	**99.87**
Corn_sensced_green_weeds	96.81	96.75	95.29	95.39	94.71	86.00	**97.19**
Lettuce_romaine_4wk	98.23	98.02	97.51	97.71	96.88	93.14	**99.69**
Lettuce_romaine_5wk	**100**	99.88	99.60	99.65	**100**	99.37	**100**
Lettuce_romaine_6wk	**99.15**	98.91	97.45	97.58	96.00	96.73	97.70
Lettuce_romaine_7wk	96.37	96.16	93.67	94.91	96.57	92.00	**97.20**
Vinyard_untrained	67.85	67.82	67.84	65.81	71.29	60.29	**78.43**
Vinyard_vertical_treils	**99.45**	99.32	98.46	98.46	98.40	94.65	94.78
OA	92.48	92.42	92.83	89.99	91.46	87.91	**94.13**
AA	96.21	96.10	96.01	94.95	95.49	91.97	**96.95**
k	**93.45**	93.27	92.00	88.85	90.49	86.51	**93.45**

Notes:

The results of SR†(Sparse Representation) and KSR†(Kernel Sparse Representation) are taken from (Zhang et al. 2016).

The results of SVM‡, OMP‡, SOMP‡ and ELM‡ are directly taken from (Li et al. 2017).

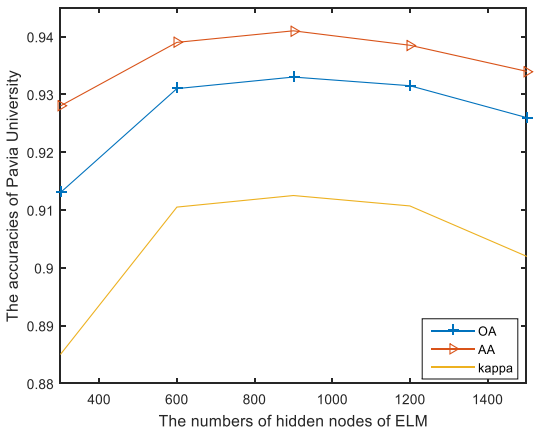
The CNN-ELM is the proposed method.

4.5 Impact of hidden neurons of ELM

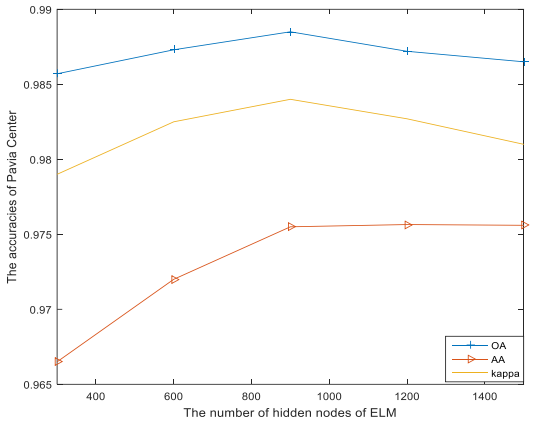
In this experiment, we conduct an evaluation of the impact of the numbers of hidden neurons of ELM using Pavia University, Pavia Center and Salinas. The number of hidden neurons of ELM is an important parameter for HSI classification, so it is worthy to discuss.

Fig. 7 (a), (b) and (c) plot the OA, AA, and kappa statistic results as a function of variable l (the numbers of hidden neurons of ELM) with 3921, 7456 and 5403 training samples for the three datasets, respectively. From Fig. 7(a), (b) and (c), we can see that l is an important parameter for HSI classification. For Pavia University and Pavia Center datasets, we should choose 900 hidden neurons. But for the Salinas dataset, we should choose 1100 hidden neurons. By choosing appropriate hidden layer nodes, we obtain the best classification accuracy for ELM. For the training samples, we choose them randomly of each class in the all labeled samples.

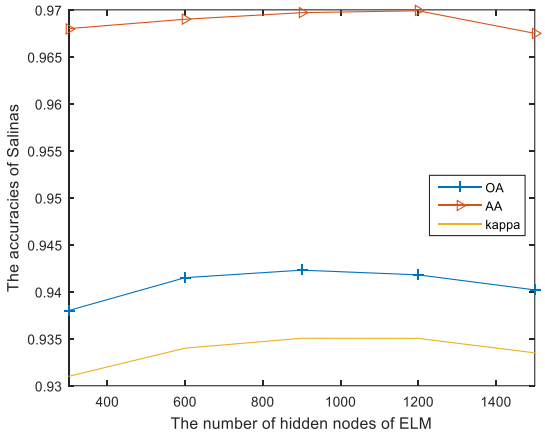
From Fig. 7 (a) and Fig. 8, we can see that the classification results are different with different l. The classification results of OA, AA, kappa statistic of Pavia University is 93.30%, 94.02%, 91.21%, respectively when the hidden neurons of ELM is set to 900, and the classification result with 900 hidden neurons of ELM outperforms other classification results with 300, 600, 1200 and 1500 hidden neurons.



(a)



(b)

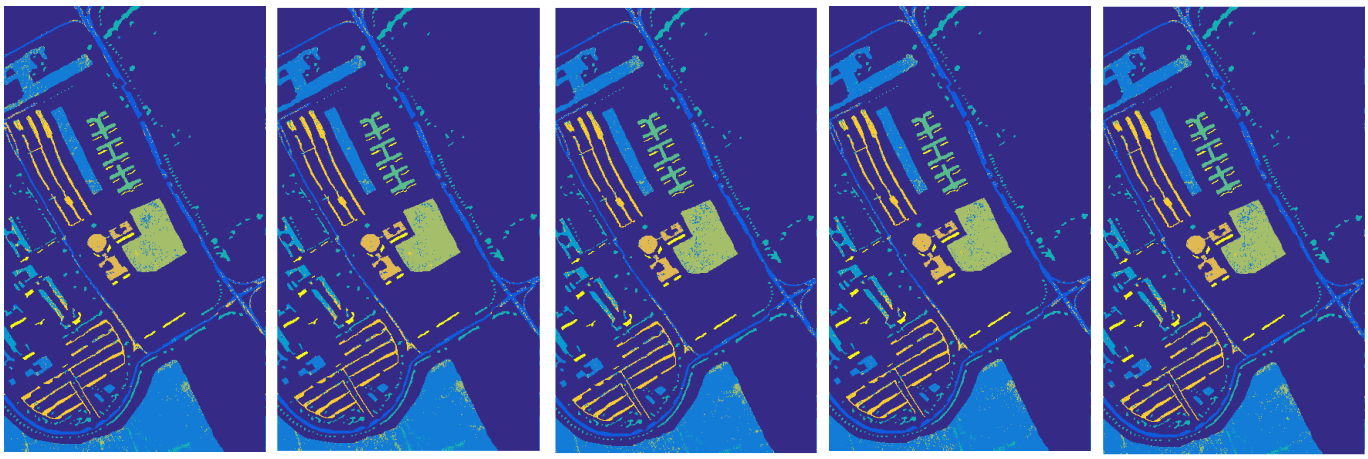


(c)

Fig. 7. The impact of hidden neurons of ELM: (a) Pavia University; (b) Pavia Center; (c) Salinas.

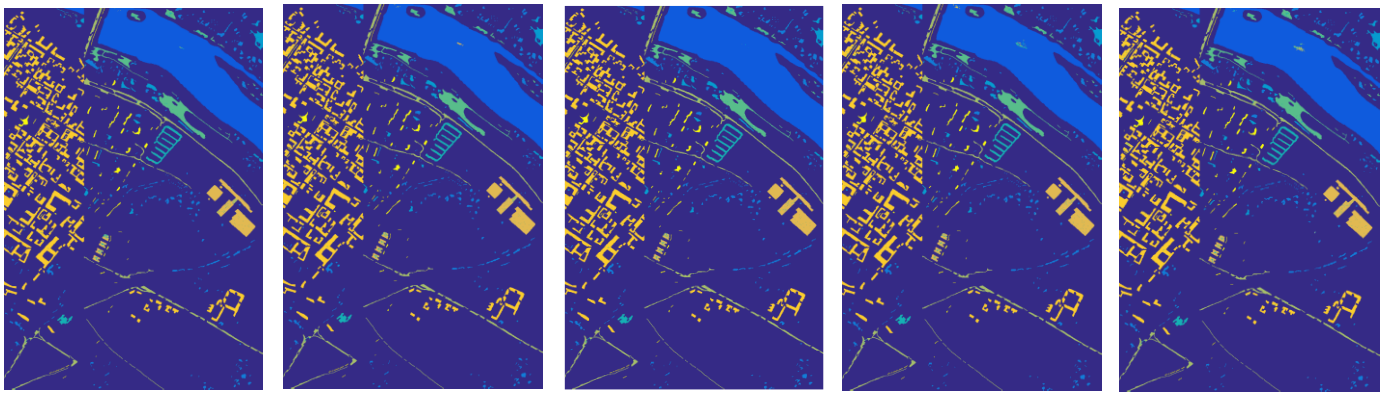
From Fig. 7 (b) and Fig. 9, although the AA of 1200 and 1500 hidden neurons are higher than 900 hidden neurons, the 900 hidden neurons achieve the best OA and kappa statistic. The OA, AA, kappa statistic with 900 hidden neurons is 98.85%, 97.50% and 98.34%, respectively. So we can say that 900 hidden neurons are the best choice for Pavia datasets.

The same as Pavia Center, from Fig. 7 (c) and Fig. 10, we can know that the AA is higher with 1400 hidden neurons than AA with 1100 hidden neurons, but the 1100 hidden neurons achieve the best classification results. So 1100 hidden neurons are the best choice for Salinas datasets.



(a) (b) (c) (d) (e)

Fig.8. The different classification results of Pavia University: (a) 300 hidden neurons of ELM with 91.27% (OA); (b) 600 hidden neurons of ELM with 93.16% (OA); (c) 900 hidden neurons of ELM with 93.3% (OA); (d) 1200 hidden neurons of ELM with 93.18% (OA); (e) 1500 hidden neurons of ELM with 92.49% (OA).



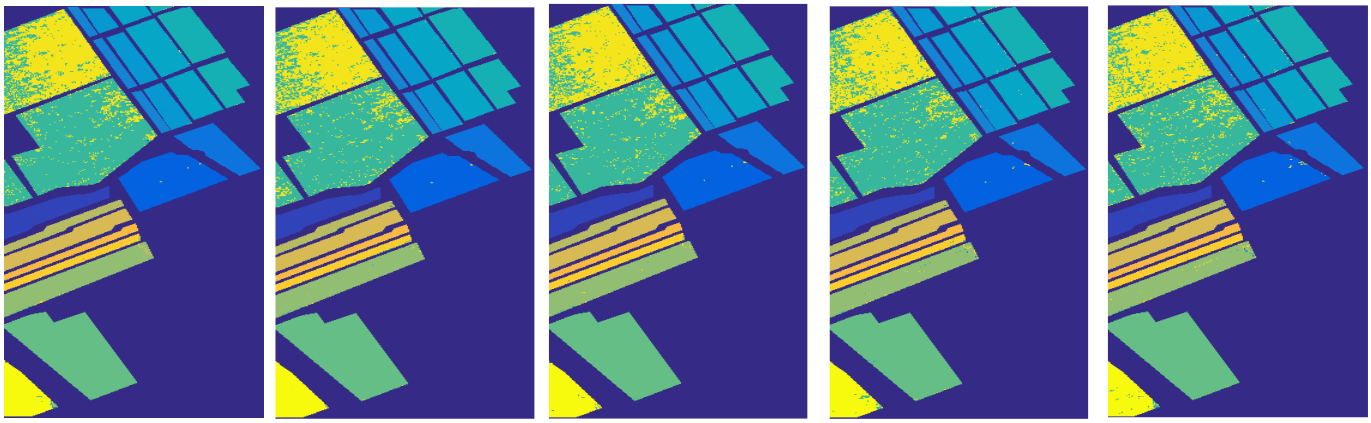
(a) (b) (c) (d) (e)

Fig.9. The different classification results of Pavia Center: (a) 300 hidden neurons of ELM with 98.57% (OA); (b) 600 hidden neurons of ELM with 98.75% (OA); (c) 900 hidden neurons of ELM with 98.85% (OA); (d) 1200 hidden neurons of ELM with 98.77% (OA); (e) 1500 hidden neurons of ELM with 98.68% (OA).

5 Conclusion

In this paper, we have proposed a new method for HSI classification by combining CNN with ELM. Firstly, we used CNN for HSI spectral feature reconstruction. Then the reconstructed feature was used for the input of ELM. Finally it was classified by ELM. This is the first time to use the reconstructing spectral of CNN as the input of ELM for HSI classification. Through the experiment results on three HSI datasets, it shows that the reconstructed spectral greatly improves the classification accuracy of HSI datasets. From the last experiment, we can see that the hidden neurons of ELM are important for HSI classification results and we achieve best results for appropriate hidden neurons.

We have improved the classification accuracy by reconstructing the spectral features, but spatial information is also important for HSI classification, so the future work will focus on using the spatial information for improve the accuracy.



(a) (b) (c) (d) (e)

Fig.10. The different classification results of Salinas: (a) 500 hidden neurons of ELM with 93.84% (OA); (b) 800 hidden neurons of ELM with 94.08% (OA); (c) 1100 hidden neurons of ELM with 94.15% (OA); (d) 1400 hidden neurons of ELM with 94.14% (OA); (e) 1700 hidden neurons of ELM with 94.00% (OA).

Acknowledgements: This work is supported by the National Nature Science Foundation of China (nos. 61471132 and 61372173), and the Training program for outstanding young teachers in higher education institutions of Guangdong Province (no. YQ2015057).

Reference

[1] Wang A, Lu J, Cai J, et al (2015) Unsupervised joint feature learning and encoding for rgb-d scene labeling. IEEE Transactions on Image Processing 24(11): 4459-4473

[2] Yu S, Jia S, Xu C (2017) Convolutional neural networks for hyperspectral image classification. Neurocomputing 219: 88-98

[3] Sun M, Zhang D, Wang Z, Ren J, Jin JS (2016) Monte Carlo convex hull model for classification of traditional Chinese paintings. Neurocomputing 171: 788-797

[4] Melgani F, Bruzzone L (2004) Classification of hyperspectral remote sensing images with support vector machines. IEEE Transactions on geoscience and remote sensing 42(8): 1778-1790

[5] Qiao T, Ren J, et al (2017) Effective denoising and classification of hyperspectral images using curvelet transform and singular spectrum analysis. IEEE Trans. Geoscience and Remote Sensing 55(1): 119-133

[6] Zabalza J, Ren J, Zheng J, Han J, Zhao H, Li S, Marshall S (2015) Novel two dimensional singular spectrum analysis for effective feature extraction and data classification in hyperspectral imaging. IEEE Trans. Geoscience and Remote Sensing 53(8): 4418-4433

[7] Qiao T, Ren J, Craigie C, Zabalza Z, Maltin C, Marshall S (2015) Singular spectrum analysis for improving hyperspectral imaging based beef eating quality evaluation. Computers and Electronics in Agriculture 115: 21-25

[8] Lu X, Wang Y, Yuan Y (2013) Graph-Regularized Low-Rank Representation for Destriping of Hyperspectral Images. IEEE Trans. Geoscience and Remote Sensing 51: 4009-4018

[9] Yuan Y, Zheng X, Lu X (2017) Discovering Diverse Subset for Unsupervised Hyperspectral Band Selection. IEEE Trans. Image Processing 26(1): 813-822

[10] Zabalza J, Ren J, Yang M, Zhang Y, Wang J, Marshall S, Han J (2014) Novel Folded-PCA for Improved Feature Extraction and Data Reduction with Hyperspectral Imaging and SAR in Remote Sensing. ISPRS Journal of Photogrammetry and Remote Sensing 93(7): 112-122

[11] Fang L, Li S, Duan W, Ren J, Atli Benediktsson J (2015) Classification of hyperspectral images by exploiting spectral-spatial information of superpixel via multiple kernels. IEEE Trans. Geoscience and Remote Sensing 53(12): 6663-6674

[12] Zabalza J, Ren J, Zheng J, et al (2016) Novel segmented stacked autoencoder for effective dimensionality reduction and feature extraction in hyperspectral imaging. Neurocomputing 185: 1-10

[13] Du Z, Li X, Lu X (2016) Local structure learning in high resolution remote sensing image retrieval. Neurocomputing 207: 813-822

[14] Ma L, Crawford MM, Tian J (2010) Local manifold learning-based-nearest-neighbor for hyperspectral image classification. IEEE Transactions on Geoscience and Remote Sensing 48(11): 4099-4109

[15] Li J, Bioucas-Dias JM, Plaza A (2012) Spectral–spatial hyperspectral image segmentation using subspace multinomial logistic regression and Markov random fields. IEEE Transactions on Geoscience and Remote Sensing 50(3): 809-823

[16] Li J, Bioucas-Dias JM, Plaza A (2010) Semisupervised hyperspectral image segmentation using multinomial logistic regression with active learning. IEEE Transactions on Geoscience and Remote Sensing 48(11): 4085-4098

[17] Huang GB, Zhu QY, Siew CK (2004) Extreme learning machine: a new learning scheme of feedforward neural networks. In: 2004 IEEE International Joint Conference on Neural Networks, pp. 985-990

[18] Bai Z, Huang GB, Wang D, et al (2014) Sparse extreme learning machine for classification. IEEE transactions on cybernetics 44(10): 1858-1870

[19] Huang GB, Zhou H, Ding X, et al (2012) Extreme learning machine for regression and multiclass classification. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 42(2): 513-529

[20] Samat A, Du P, Liu S, et al (2014) Ensemble Extreme Learning Machines for Hyperspectral Image Classification. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 7(4): 1060-1069

[21] Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. Science, 313(5786): 504-507

[22] Hu W, Huang Y, Wei L, et al (2015) Deep convolutional neural networks for hyperspectral image classification. Journal of Sensors 2015: 1-12.

[23] Fukushima K (1988) Neocognitron: A hierarchical neural network capable of visual pattern recognition. Neural networks 1(2): 119-130

[24] Lecun Y, Bottou L, Bengio Y, et al (1998) Gradient-based learning applied to document recognition. Proceedings of the IEEE 86(11): 2278-2324

[25] Ciresan DC, Meier U, Masci J, et al (2011) Flexible, high performance convolutional neural networks for image classification. In: IJCAI Proceedings-International Joint Conference on Artificial Intelligence, pp. 1237-1242

[26] Simard PY, Steinkraus D, Platt JC (2003) Best practices for convolutional neural networks applied to visual document analysis. In: ICDAR '03 Proceedings of the Seventh International Conference on Document Analysis and Recognition, pp. 958-962

[27] Slavkovikj V, Verstockt S, et al (2015) Hyperspectral image classification with convolutional neural networks. In: Proceedings of the 23rd ACM international conference on Multimedia, pp. 1159-1162

[28] Yue J, Zhao W, Mao S, et al (2015) Spectral–spatial classification of hyperspectral images using deep convolutional neural networks. Remote Sensing Letters 6(6): 468-477

[29] Makantasis K, Karantzalos K, Doulamis A, et al (2015) Deep supervised learning for hyperspectral data classification through convolutional neural networks. In: 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 4959-4962

[30] Huang GB, Wang DH, Lan Y (2011) Extreme learning machines: a survey. International Journal of Machine Learning and Cybernetics 2(2): 107-122

[31] Sainath TN, Mohamed A, Kingsbury B, et al (2013) Deep convolutional neural networks for LVCSR. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, pp.8614-8618

[32] Lv Q, Niu X, Dou Y, et al (2016) Classification of hyperspectral remote sensing image using hierarchical local-receptive-field-based extreme learning machine. IEEE Geoscience and Remote Sensing Letters 13(3): 434-438

[33] Sun L, Wu Z, Liu J, et al (2015) Supervised spectral–spatial hyperspectral image classification with weighted Markov random fields. IEEE Transactions on Geoscience and Remote Sensing 53(3): 1490-1503

[34] Plaza A, Benediktsson JA, Boardman JW, et al (2009) Recent advances in techniques for hyperspectral image processing. Remote sensing

of environment 113: 110-122

[35] Tarabalka Y, Chanussot J, Benediktsson JA (2010) Segmentation and classification of hyperspectral images using watershed transformation. Pattern Recognition 43(7): 2367-2379

[36] Li J, Bioucas-Dias JM, Plaza A (2013) Spectral–spatial classification of hyperspectral data using loopy belief propagation and active learning. IEEE Transactions on Geoscience and Remote Sensing 51(2): 844-856

[37] Ghamisi P, Benediktsson JA, Sveinsson JR (2014) Automatic spectral–spatial classification framework based on attribute profiles and supervised feature extraction. IEEE Transactions on Geoscience and Remote Sensing 52(9): 5771-5782

[38] Li Y, Xie W, Li H (2017) Hyperspectral image reconstruction by deep convolutional neural network for classification. Pattern Recognition 63: 371-383

[39] Zhang X, Liang Y, Zheng Y, et al (2016) Hierarchical Discriminative Feature Learning for Hyperspectral Image Classification. IEEE Geoscience and Remote Sensing Letters 13(4): 594-598.

[40] Li W, Chen C, Su H, et al. (2015) Local binary patterns and extreme learning machine for hyperspectral imagery classification. IEEE Transactions on Geoscience and Remote Sensing, 53: 3681-3693.

[41] Hughes, G (1968) .On the mean accuracy of statistical pattern recognizers. IEEE Transactions on Information. Theory. 14: 55-63.

[42] Khan M J, Yousaf A, Javed N, et al (2017). Automatic Target Detection in Satellite Images using Deep Learning. Journal of Space Technology, 7(1).

[43] Leng J, Li T, Bai G, et al (2016). Cube-CNN-SVM: A Novel Hyperspectral Image Classification Method//Tools with Artificial Intelligence (ICTAI), 2016 IEEE 28th International Conference on. IEEE, 1027-1034.

[44] Khan M J, Khan H S, Yousaf A, et al (2018). Modern Trends in Hyperspectral Image Analysis: A Review. IEEE Access, 6: 14118-14129.

[45] Boldrini B, Kessler W, Rebner K, et al (2012). Hyperspectral imaging: a review of best practice, performance and pitfalls for in-line and on-line applications. Journal of near infrared spectroscopy, 20(5): 483-508.

Author biography

Faxian Cao:

Faxian Cao was born in Jiangxi, China, in September 1993. He received the B.E. degree from school of electronic and information engineering at Jinggangshan University, Ji'an, China, in 2016. Now, he is currently pursuing his master degree in school of information engineering at Guangdong University of Technology. His main research interests comprise Extreme Learning Machine, Pattern Recognition; Hyperspectral image classification, feature extraction and denoising.

Zhijing Yang:

Dr. Zhijing Yang received the B.S and Ph.D. degrees from the Mathematics and Computing Science, Sun Yat-Sen University, Guangzhou China, in 2003 and 2008, respectively. He is currently an Associate Professor in the Scholl of Information Engineering, Guangdong University of Technology, China. His research interests include time-frequency analysis, signal processing, machine learning, and pattern recognition.

Jinchang Ren:

Dr. Jinchang Ren received the B.E degree in computer software, the M.Eng degree in image processing, and the D.Eng degree in computer vision from Northwestern Polytechnical University, Xi'an, China, and the Ph.D degree in electronic imaging and media communication from Bradford University, Bradford, U.K. His research interests are focused mainly on semantic content extraction for video analysis and understanding and, more recently, hyperspectral imaging.

Wing-Kuen Ling:

Prof. Ling obtained his B. Eng. degree and M. Phil. degree from the Hong Kong University of Science and Technology in 1997 and 2000, respectively, and the Ph. D. degree from the Hong Kong Polytechnic University in 2003. His research interests include optimization theory, time frequency analysis, biomedical signal processing, multimedia signal processing, nonlinear digital signal processing systems, and control theory.

Figure Captions

Fig. 1. The architecture of an ELM.

Fig. 2. A typical architecture of CNN consists of input layer, convolutional layer, max pooling layer and fully connected layer.

Fig.3 The flowchart of the proposed CNN-ELM.

Fig. 4. Pavia University dataset: (a) Training samples; (b) Testing classification results.

Fig. 5. Pavia Center dataset: (a) Training samples; (b) Testing classification results.

Fig. 6 Salinas dataset: (a) Training samples; (b) Testing classification results.

Fig. 7. The impact of hidden neurons of ELM: (a) Pavia University; (b) Pavia Center; (c) Salinas.

Fig.8. The different classification results of Pavia University: (a) 300 hidden neurons of ELM with 91.27% (OA); (b) 600 hidden neurons of ELM with 93.16% (OA); (c) 900 hidden neurons of ELM with 93.3% (OA); (d) 1200 hidden neurons of ELM with 93.18% (OA); (e) 1500 hidden neurons of ELM with 92.49% (OA).

Fig.9. The different classification results of Pavia Center: (a) 300 hidden neurons of ELM with 98.57% (OA); (b) 600 hidden neurons of ELM with 98.75% (OA); (c) 900 hidden neurons of ELM with 98.85% (OA); (d) 1200 hidden neurons of ELM with 98.77% (OA); (e) 1500 hidden neurons of ELM with 98.68% (OA).

Fig.10. The different classification results of Salinas: (a) 500 hidden neurons of ELM with 93.84% (OA); (b) 800 hidden neurons of ELM with 94.08% (OA); (c) 1100 hidden neurons of ELM with 94.15% (OA); (d) 1400 hidden neurons of ELM with 94.14% (OA); (e) 1700 hidden neurons of ELM with 94.00% (OA).