Journal of Real-Time Image Processing
https://doi.org/10.1007/s11554-018-0793-9

SPECIAL ISSUE PAPER

Fast dimensionality reduction and classification of hyperspectral images with extreme learning machines

Juan Mario Haut[1] · Mercedes Eugenia Paoletti[1] · Javier Plaza[1] · Antonio Plaza[1]

Received: 30 December 2017 / Accepted: 24 May 2018

Abstract
Recent advances in remote sensing techniques allow for the collection of hyperspectral images with enhanced spatial and spectral resolution. In many applications, these images need to be processed and interpreted in real-time, since analysis results need to be obtained almost instantaneously. However, the large amount of data that these images comprise introduces significant processing challenges. This also complicates the analysis performed by traditional machine learning algorithms. To address this issue, dimensionality reduction techniques aim at reducing the complexity of data while retaining the relevant information for the analysis, removing noise and redundant information. In this paper, we present a new real-time method for dimensionality reduction and classification of hyperspectral images. The newly proposed method exploits artificial neural networks, which are used to develop a fast compressor based on the extreme learning machine. The obtained experimental results indicate that the proposed method has the ability to compress and classify high-dimensional images fast enough for practical use in real-time applications.

Keywords Hyperspectral imaging · Real-time processing · Extreme learning machine · Dimensionality reduction · Classification

1 Introduction

Current Earth observation (EO) missions utilize sensors that allow for the acquisition of high-dimensional image data with enhanced spectral and high spatial resolution. Specifically, imaging spectrometers collect information for large areas on the surface of the Earth with a high spectral resolution and narrow bands, while multispectral images generally contain higher spatial resolution and a lower number of spectral bands [1]. Hyperspectral images record data throughout the visible and solar-reflected infrared spectrum at different wavelength channels [2–4]. As a result, three-dimensional data cubes are generated with size significantly larger than traditional panchromatic (PAN) or RGB images, allowing to solve problems which usually cannot be solved by these images. However, the large amount of information contained in these images introduces important challenges, both from the viewpoint of storage and processing requirements. In addition, these data are characterized by the presence of redundant information (in particular, some spectral bands may be highly correlated and present some redundancies), in addition to spatial variability and noise. These data dimensionality issues, coupled with the increasing complexity of remote sensing hardware and software, introduced the need to develop new data analysis and compression methods that allow for faster information extraction, thus increasing product reliability.

Reduced data representations aim at exploring the minimum number of parameters needed to retain the original

This work has been supported by Ministerio de Educación (Resolución de 26 de diciembre de 2014 y de 19 de noviembre de 2015, de la Secretaría de Estado de Educación, Formación Profesional y Universidades, por la que se convocan ayudas para la formación de profesorado universitario, de los subprogramas de Formación y de Movilidad incluidos en el Programa Estatal de Promoción del Talento y su Empleabilidad, en el marco del Plan Estatal de Investigación Científica y Técnica y de Innovación 2013–2016. This work has also been supported by Junta de Extremadura (decreto 297/2014, ayudas para la realización de actividades de investigación y desarrollo tecnológico, de divulgación y de transferencia de conocimiento por los Grupos de Investigación de Extremadura, Ref. GR15005).

✉ Juan Mario Haut
juanmariohaut@unex.es

[1] Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, University of Extremadura, Caceres, Spain

properties of the data [5]. The analysis of hyperspectral images is generally affected by the Hughes' phenomenon [6]. For instance, the accuracy of hyperspectral classification techniques decreases as the dimensionality of the input data increases, depending on the training samples and the structure of the classifier used. In fact, the imbalance between the high spectral dimensionality of hyperspectral data and the limited number of available training samples can produces the processing method to fall into overfitting, reducing its ability to generalize (curse of dimensionality [7]). As a result, dimensionality reduction (DR) techniques have been widely used prior to classification of hyperspectral images. Traditionally, DR is performed using well-known linear techniques such as principal component analysis (PCA) [8–10], which calculates the projection according to which the data are better represented in terms of least squares. Other widely used PCA based techniques include kernel PCA (KPCA) [11] and distributed-parallel PCA (PCA-DP) [12]. Similarly, the Non-Negative Matrix Factorization (NMF) has been used as feature extraction algorithm, removing redundant information and reducing processing dimensions while taking into account nonnegativity of the data [13, 14] Other popular DR techniques such as independent component analysis (ICA) [15, 16] have shown success in fields such as blind source separation, feature extraction and unsupervised recognition, but can also be used for data reduction and compression (ICA-DR) [17]. The maximum noise fraction (MNF) transformation [18] (also called noise-adjusted principal components transform or NAPC [19, 20]) has been useful to segregate noise in hyperspectral data [21], normalizing the linear combinations of original bands by maximizing the signal-to-noise ratio (SNR). To overcome the disadvantages that MNF presents, the interference and noise-adjusted principal components analysis (INAPCA) [19] was proposed with good results. Based on signal subspace identification (SSI), other DR approaches take into account the preservation of rare pixels that are scarcely represented in the hyperspectral scene and contain linearly independent spectral features, such as the redundancy reduction approach, also called maximum orthogonal-complements algorithm (MOCA) [22] or the robust signal subspace estimation (RSSE) [23], which derives the signal subspace and estimates the data dimensionality, both studied in [24]. Following previous works, [25] proposed a new signal subspace identification method for DR, which takes into account both the abundant and the rare signal components, developing a new subspace estimation criterion.

In recent years, the use of artificial neural networks (ANNs) for the processing of remote sensing data in general [26] and hyperspectral images in particular [27, 28] has gained much popularity, with promising results in the field of pattern recognition [29]. An interesting application of ANNs is data compression/decompression, which can be performed using autoassociative neural networks (AANNs), also called autoencoders [30–35]. These ANN architectures have been quite popular for data denoising and DR. Autoencoders are simple unsupervised learning networks whose aim is to project the original inputs into a new space, i.e., to generate compressed or extended outputs with the least possible amount of distortion. With appropriate dimensionality and sparsity constraints, autoencoders can learn data projections that are more effective than those provided by other techniques such as PCA, allowing for the design of non-linear PCAs (NLPCAs) [36, 37]. Additionally, autoencoders can be designed from shallow to deep neural networks (DNNs), being the multi-layer perceptron (MLP) [31, 38] with three layers (input–hidden–output) the most widely used implementation. ANN-based DR methods offer an attractive possibility, due to the fact that they do not need prior knowledge on the statistical distribution of the classes and can take advantage of multiple training techniques to deal with linearly non-separable data [39, 40]. However, these methods have been traditionally affected by its algorithmic and computational complexity [41]. This fact becomes critical in real-time remote sensing applications [42, 43], where processing algorithms must support immediate decision-making in critical circumstances [44] or in scenarios in which communicating the data from the airborne/satellite sensor to the processing ground station is necessary.

To deal with the high computational cost of autoencoder-based DR methods, this paper presents a new fast method for real-time dimensionality reduction (compression) and classification of hyperspectral images. This is an important contribution, since the proposed approach can deal with several problems of remotely sensed data exploitation. First and foremost, the proposed approach can perform real-time onboard data compression, thus eliminating the need to transmit the data to the processing ground station before performing the analysis. Most importantly, the newly developed approach can exploit ANN-based methods to perform real-time classification of hyperspectral data. For this purpose, we resort to the extreme learning machine (ELM) architecture [45]. ELMs can be considered as a promising learning algorithm, whose implementation is based on the minimization of the training error and the norm of the output weights. In this way, ELMs are computational efficient because only the network architecture is fine-tuned, with no additional trainable parameters, reaching a high generalization performance [46]. They have been used for hyperspectral processing before, performing a wide range of applications, such as classification, clustering, ranking and compression. For classification purposes, we can find several interesting works in the literature, for example in [47] the single layer ELM has been developed for both, multispectral and hyperspectral remote sensing image classification, providing a final accuracy comparable to backpropagation neural network. Different ELM models have been also used for hyperspectral classification in combination with kernel methods [48–53], differential evolution [54], ensemble learning [46], among other elaborated approaches [55–57]. Also, ELMs can be used for unsupervised clustering [58] and ranking [59] tasks. On the other part, several works have been carried out on hyperspectral's DR with ELM by selecting a subset of original bands (band selection) [60]. We must remark the lack of works on hyperspectral DR and classification with ELM in literature. In fact, research seems to focus on the hyperspectral classification with ELM after a pre-processing data reduction with methods such as PCA or Firefly algorithm [61–63]. In this sense, this work proposes a global model to hyperspectral reduction and classification based on ELM. In summary, the main contribution of our work is the development of a new real-time method for DR and classification of hyperspectral images that exploits ANNs to develop a fast compressor based on the ELM architecture [64–66].

The remainder of the paper is organized as follows. First, we present some basic concepts about the PCA as a standard approach for DR. Also, the ICA and the NMF are presented as DR methods for hyperspectral compression, together with the main concept behind autoencoders and the ELM algorithm. Then, we describe our newly developed fast DR and classification method. Next, we provide an extensive experimental evaluation, evaluating the ability of the proposed method to compress and classify hyperspectral images fast enough for practical exploitation in real-time. The paper concludes with some remarks and hints at plausible future lines.

2 Related work

2.1 Principal component analisys (PCA)

Let us assume a set of N observations, where each sample is given by $\mathbf{x}_i = [x_{i,1}, x_{i,2}, ..., x_{i,d}]^T \in \mathbb{R}^d$, with $i = 1, ..., N$, contained in a matrix $X \in \mathbb{R}^{d\times N}$, where the i-th column represents the sample $\mathbf{x}_i$ and the d-th row represents the d-th feature of sample $\mathbf{x}_i$ so that

$$X = \begin{pmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_N \end{pmatrix}^T = \begin{pmatrix} x_{1,1} & \cdots & x_{1,d} \\ \vdots & \ddots & \vdots \\ x_{N,1} & \cdots & x_{N,d} \end{pmatrix}^T$$

The goal of PCA as a statistical technique for unsupervised DR is to find the subspace that captures most variance in data, extracting the underlying structure of the matrix X, i.e., the principal components understood as directions with largest variance, where the data is most spread out [8, 9], to reduce X down to its basic components. In fact, PCA is an orthogonal transformation that converts the observations represented by matrix $X \in \mathbb{R}^{d\times N}$ into a matrix $X'_i \in \mathbb{R}^{l\times N}$, whose vectors $\mathbf{x}'_i = [x'_{i,1}, x'_{i,2}, ..., x'_{i,l}]^T \in \mathbb{R}^l$ have smaller dimension, $l < d$. Each vector $\mathbf{x}'_i$ contains the set of linearly uncorrelated variables or principal components of $\mathbf{x}_i$. The underlying transformation is defined by the linear Eq. 1:

$$X' = WX \tag{1}$$

where $W \in \mathbb{R}^{l\times d}$ is the projection matrix created by the l selected eigenvectors, $\boldsymbol{v}$, of the input data covariance matrix. To do that, PCA standardizes the input data X along the N columns creating matrix Y whose columns $\mathbf{y}_i$ are calculated as:

$$\mathbf{y}_i = \mathbf{x}_i - \frac{1}{N}\sum_{j=1}^{N} x_{i,j} \quad i = 1, ..., d \tag{2}$$

From matrix Y, PCA calculates the covariance matrix $\text{cov}(Y)$ and extracts the eigenvectors $\boldsymbol{v}$ and eigenvalues λ (expressed as a diagonal matrix) as $\boldsymbol{v}^{-1}\text{cov}(Y)\boldsymbol{v} = \lambda$. Eigenvectors and eigenvalues are pointing in the direction of maximal variance with the corresponding magnitude of the variance, respectively. Then, eigenvalues can be sorted in descending order, selecting the first l eigenvectors (being l the new dimension of the features) to create the transformation matrix W, whose rows are orthogonal to the preceding ones. Finally, PCA projects the input data X from space $\mathbb{R}^{d\times N}$ to the output data X' in space $\mathbb{R}^{l\times N}$ using equation 1.

On the other hand, to recover the original input data dimensions, PCA can make the reverse process, i.e., $X'' \approx W^T X'$, to invert the formula and get an approximation to original data. The goal is to minimize the difference between the original data X and the reconstructed data X'', so that its objective function will be $\min \parallel X'' - X \parallel_2$ that can be also represented as:

$$\min_W \parallel W^T(WX) - X \parallel_2 \tag{3}$$

2.2 Independent component analysis (ICA)

ICA algorithm has been used traditionally as an unsupervised statistical technique for blind source separation, separating a multivariate/superimposed signal into additive subcomponents, called independent component (ICs), i.e., its goal is to find the subspace where sources are independent. To do that, it assumes that those ICs are non-Gaussian signals and they are statistically independent from each other [67–69]. The problem can be defined as follows: supposing that $\mathbf{s}(t)$ is an unobserved-original n-dimensional signal obtained at time t, $\mathbf{s}(t) \in \mathbb{R}^n = [s_1(t), s_2(t), ..., s_n(t)]^T$, where each component is statistically independent, and it is mixed into the m-dimensional observed vector signal $\mathbf{x}(t) \in \mathbb{R}^m = [x_1(t), x_2(t), ..., x_m(t)]^T$ via an unknown matrix $A_{m\times n}$ as:

$$\mathbf{x}(t) = A \cdot \mathbf{s}(t) \tag{4}$$

The goal of ICA is to find the unmixing matrix $W_{n \times m}$ to approximately recover the original source $\mathbf{s}'(t) \approx \mathbf{s}(t)$, separating $\mathbf{x}(t)$ into a set of sources which are statistically independent, such as:

$$\mathbf{s}'(t) = W \cdot \mathbf{x}(t) \tag{5}$$

In recent years ICA has been exploited also in remote sensing image processing. In particular, ICA has proved to be very useful in hyperspectral image analysis [70, 71], extracting relevant spectral information for classification [72, 73], dimensionality reduction [17] and unmixing problems [74].

Focusing on DR, hyperspectral data can be interpreted as a mixture of signals where ICA algorithm can be applied as a feature extraction method to provide ICs which can extract information related to one or more classes [70]. In this case, FastICA [75] has been employed as hard non-convex optimization method. Let us assume the input data matrix $X \in \mathbb{R}^{d \times N}$, where N is the number of samples of mixed signals, i.e., each $\mathbf{x}_i \in \mathbb{R}^d = [x_{i,1}, x_{i,2}, ..., x_{i,d}]^T$ with $i = \{1, 2, ..., N\}$, and d the number of independent source signals. After centering and whitening of the data via PCA (pre-whitening step so that components are uncorrelated), ICA extracts the l independent components, also considered as projection pursuit directions, from the input matrix X using l units (e.g., neurons) with weight vectors $\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_l$. Algorithm 1 shows the followed process, where $g(u) = \tanh(u)$ is the first derivative of $\mathrm{f}(u) = \log\cosh(u)$ and $g'(u) = 1 - \tanh^2(u)$ is the second derivative of $\mathrm{f}(u)$. In our case, the parallel FastICA has been implemented, with a tolerance value set to 1e−04 and 200 as maximum number of iterations.

Algorithm 1 FastICA algorithm for multiple component extraction

```
1: procedure FastICA(l,X)    ▷ $l$: number of desired ICs, $l \le d$, $X_{d \times N}$: input prewhitened matrix
2:     for $i = 1$ to $l$ do
3:         $\mathbf{w}_i \leftarrow$Random vector of length $d$
4:         while $\mathbf{w}_i$ changes AND $tol$ <threshold do
5:             $\mathbf{w}_i \leftarrow \frac{1}{N} X \mathrm{g}\left(\mathbf{w}_i^T X\right)^T - \frac{1}{N} \mathrm{g}'\left(\mathbf{w}_i^T X\right) \mathbf{w}_i$
6:             $\mathbf{w}_i \leftarrow \mathbf{w}_i - \left(\sum_{j=1}^{i-1} \mathbf{w}_i^T \mathbf{w}_j \mathbf{w}_j^T\right)^T$
7:             $\mathbf{w}_i \leftarrow \frac{\mathbf{w}_i}{\|\mathbf{w}_i\|}$
8:     $W = [\mathbf{w}_1, \mathbf{w}_2, \cdots \mathbf{w}_l]$    ▷ $W_{d \times l}$: unmixing matrix
9:     $S = WX$    ▷ $S_{l \times N}$: independent components matrix
10:    Return $W, S'$
```

2.3 Non-negative matrix factorization (NMF)

The NMF is a popular linear dimensionality reduction and noise whitening method [76] whose goal is to find a subspace that minimizes reconstruction error assuming that the data and the components are non-negative. Assuming $X \in \mathbb{R}^{d \times N}$ is the input data matrix, where d is the number of variables and N is the number of observations, i.e., each sample $\mathbf{x}_i \in \mathbb{R}^d = [x_{i,1}.x_{i,2}, ..., x_{i,d}]^T$ with $i = \{1, 2, ..., N\}$, the goal of NMF is to decompose X into two matrices $W \in \mathbb{R}^{d \times l}$ and $H \in \mathbb{R}^{l \times N}$ of non-negative elements, with W and H smaller than X because of rank value l, optimizing the distance d between X and WH, i.e.

$$X \simeq WH = \begin{pmatrix} x_{1,1} & \cdots & x_{1,N} \\ \vdots & \ddots & \vdots \\ x_{d,1} & \cdots & x_{d,N} \end{pmatrix} \simeq \begin{pmatrix} w_{1,1} & \cdots & w_{1,l} \\ \vdots & \ddots & \vdots \\ w_{d,1} & \cdots & w_{d,l} \end{pmatrix} \begin{pmatrix} h_{1,1} & \cdots & h_{1,N} \\ \vdots & \ddots & \vdots \\ h_{l,1} & \cdots & h_{l,N} \end{pmatrix} \tag{6}$$

$$\text{with } \mathbf{x}_i = \sum_{j=1}^{l} h_{j,i} \cdot \mathbf{w}_j$$

That means, each columns of X, represented by $\mathbf{x}_i$, can be computed as linear combination of column vectors of W (considered as basis vectors of the vector space defined by X) and the positive coefficients of H, $h_{j,i}$ by minimizing d, which can be defined as the squared Frobenius norm (in this case NMF works similar to PCA, but with non-negativity constraints) or as the Kullback–Leibler (KL) divergence.

$$d = \frac{1}{2} \parallel X - WH \parallel_F^2 = \sum_{i=1}^{N} \sum_{j=1}^{d} \left(x_{j,i} - (w \cdot h)_{j,i}\right)^2 \tag{7}$$

In our case, the squared Frobenius norm (Eq. 7) has been used as objective function with the Coordinate Descent solve as optimizer, whereas NMF's pair of factors (W, H) has been initialized by Nonnegative Double Singular Value Decomposition (NNDSVD) [77], which is better fit for sparse factorization. On the other part the tolerance has set to 1e−04, while the maximum of iterations has been set to 200.

2.4 Autoencoder

Similar to PCA, the autoencoder is an unsupervised learning algorithm that also minimizes the same objective function as PCA (see Eq. 3). However, the autoencoder is more flexible than the PCA. This is because: (1) the autoencoder manages non-linearities in the encoding phase using non-linear activation functions (e.g., sigmoid, hyperbolic tangent or ReLU, among others), whereas the PCA can only represent linear transformations; and (2) the autoencoder offers the possibility of using stack layers to create deeper architectures.

Let X be a set of N samples $X = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$, where each $\mathbf{x}_i \in \mathbb{R}^d$ so $\mathbf{x}_i = [x_{i,1}, x_{i,2}, ..., x_{i,d}]^T$. The goal of the autoencoder is to represent (or to project) the original inputs in $\mathbb{R}^d$ into a new space $\mathbb{R}^l$. So, the basic operation of an autoencoder is quite intuitive: given the input vector,

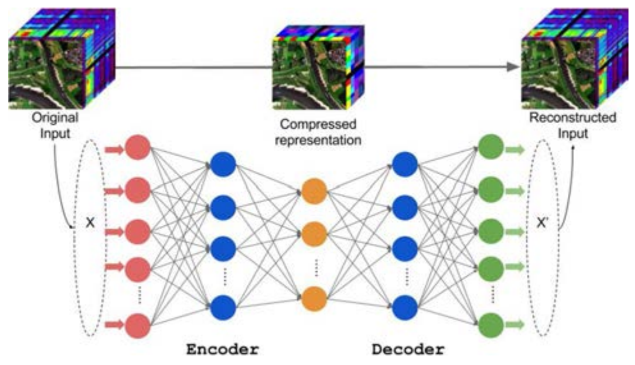


Fig. 1 Encoder-decoder functions of a simple autoencoder

$\mathbf{x}_i \in \mathbb{R}^d$, the autoencoder network uses a set of recognition weights (encoder components) to convert $\mathbf{x}_i$ into a code vector or code dictionary $\mathbf{c}_i \in \mathbb{R}^l$, so $\mathbf{c}_i = [c_{i,1}, c_{i,2}, ..., c_{i,l}]^T$ (mapping $\mathbb{R}^d$ space to $\mathbb{R}^l$ space). To recover the original input $\mathbf{x}_i \in \mathbb{R}^d$, it uses a set of generative weights (decoder component) to convert the code vector into an approximate reconstruction of the input vector, $\mathbf{x}'_i \in \mathbb{R}^d$ (de-mapping $\mathbb{R}^l$ space to $\mathbb{R}^d$ space) [30].

As we can see in Fig. 1, a simple autoencoder is composed by two networks, the compressor or encoder network, and the decompressor or decoder network, together with a middle-hidden layer that determines the desired compression ratio of the autoencoder. The code vector $\mathbf{c}_i \in \mathbb{R}^l$ corresponding to the input $\mathbf{x}_i \in \mathbb{R}^d$ and the autoencoder reconstruction output $\mathbf{x}'_i \in \mathbb{R}^d$ are calculated as follows:

$$\begin{aligned} \mathbf{c}_i &= f(W \cdot \mathbf{x}_i + \mathbf{b}), \\ \mathbf{x}'_i &= f(W^T \cdot \mathbf{c}_i + \mathbf{b}'), \end{aligned} \tag{8}$$

where $f(\cdot)$ is the activation function, W is the weight matrix of the encoder, W^T is the tied weight matrix of the decoder (although decoder can use an untied weight matrix, separating the encoder from the decoder), and $\mathbf{b}$ and $\mathbf{b}'$ are the typical bias term. The autoencoder learning process occurs through the back-propagation of the reconstruction error:

$$\min \parallel \mathbf{x}_i - \mathbf{x}'_i \parallel_2 \tag{9}$$

An autoencoder can be simply implemented with a MLP network [31, 38], from shallow architectures to deep learning models.

2.5 Extreme learning machine (ELM)

The ELM is a relatively recent learning algorithm [45, 78, 79] where the neural network learning parameters (input weights W and biases B) are randomly assigned and do not need to be tuned, while the final output weights (β) can be analytically determined by a simple generalized inverse operation [80], without the need for validation or human-intervened parameters [78]. These characteristics make the ELM a powerful and easy tool, able to provide efficient-unified solutions in data regression and classification [81], mainly due to its extremely fast learning speed and great generalization capability in a wide range of applications [82], including clustering [83, 84], regression [66, 78] and classification [81] in semi-supervised, supervised and unsupervised fashion [85].

Although the ELM has obtained good results in hierarchical learning with deep architectures [64, 65, 82, 86–88], the original ELM was based on single-hidden layer feed-forward neural networks (SLFNs) [66, 78, 79, 89–92], whose hidden layer does not need to be tuned. The classical ELM model (see Fig. 2) consists of three steps:

1. First, it calculates and assigns random weights W and biases B between the input layer and the hidden layer. W and B are generated based on a continuous sampling distribution probability. It has been proved that SLFNs with a hidden layer and L random hidden neurons can exactly learn L distinct observations [93, 94], so it is not strictly necessary to adjust the input weights and first hidden layer biases if later the network can adjust these L representations to the desired output [45].
2. Then, it calculates the hidden layer's output matrix, H, that maps the data to the L-dimensional hidden layer random feature space (ELM feature function: $H : \mathbb{R}^d \to \mathbb{R}^L$).
3. Finally, after the random nonlinear feature mapping, the rest of the ELM can be considered as a linear system [85, 95] that calculates the connection weights between the hidden layer and the output layer, β that best approximate the output of the hidden layer to the desired output of the network while exhibiting the minor norm at the same time [81].

Given N training samples and their desired outputs $\{(\mathbf{x}_i, \mathbf{t}_i)\}_{i=1}^N$, where each $\mathbf{x}_i = [x_{i,1}, x_{i,2}, ..., x_{i,d}]^T \in \mathbb{R}^d$ and each

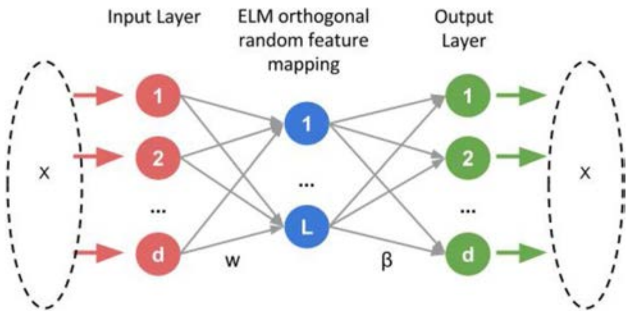


Fig. 2 Extreme learning machine scheme

$\mathbf{t}_i \in \mathbb{R}^m$ so $\mathbf{t}_i = [t_{i,1}, t_{i,2}, ..., t_{i,m}]^T$, the output function of the ELM for the generalized SLFN, $\mathbf{t}_i = f(\mathbf{x}_i)$, can be defined as:

$$f(\mathbf{x}_i) = \sum_{j=1}^{L} \boldsymbol{\beta}_j \cdot h_j(\mathbf{x}_i) \quad i = 1, 2, ..., n, \tag{10}$$

where $\boldsymbol{\beta}_j = [\beta_{j,1}, \beta_{j,2}, ..., \beta_{j,m}]^T \in \mathbb{R}^m$ is the weight vector that connects the j-th hidden neuron with the output nodes and $h_j(\mathbf{x}_i) = h(\mathbf{w}_j \cdot \mathbf{x}_i + b_j)$ is the output of the j-th neuron in the hidden layer, with $\mathbf{w}_j \in \mathbb{R}^d$ so $\mathbf{w}_j = [w_{j,1}, w_{j,2}, ..., w_{j,d}]^T$ is the random weight vector that connects the j-th node with input nodes and b_j contains the random bias or threshold of the hidden neuron j. Eq. 10 can be compacted as:

$$f(X) = H\beta = T, \tag{11}$$

where $H_{N\times L}$ is the output matrix of the hidden layer, calculated as the inner product between the input matrix data $X_{N\times d} = \left(\mathbf{x}_1^T, \mathbf{x}_2^T, ..., \mathbf{x}_N^T\right)^T$ and the matrix of the hidden layer weights $W_{d\times L} = \left(\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_L\right)$, to which the bias is added and the $h(\cdot)$ is passed as a nonlinear piecewise continuous function, such as the sigmoid, Gaussian, multiquadrics, Fourier or Hardlimit functions:

$$H(W,B,X) \rightarrow X \cdot W + \mathbf{b} =$$

$$\begin{pmatrix} x_{1,1} & \cdots & x_{1,d} \\ \vdots & \ddots & \vdots \\ x_{N,1} & \cdots & x_{N,d} \end{pmatrix} \cdot \begin{pmatrix} w_{1,1} & \cdots & w_{1,d} \\ \vdots & \ddots & \vdots \\ w_{L,1} & \cdots & w_{L,d} \end{pmatrix}^T + \begin{pmatrix} b_1 \\ \vdots \\ b_L \end{pmatrix}^T$$

So $H(W,B,X) = H(\mathbf{w}_1, ..., \mathbf{w}_L, b_1, ..., b_L, \mathbf{x}_1, ..., \mathbf{x}_N)$ contains:

$$H = \begin{pmatrix} h(\mathbf{w}_1\mathbf{x}_1 + b_1) & ... & h(\mathbf{w}_L\mathbf{x}_1 + b_L) \\ \vdots & \ddots & \vdots \\ h(\mathbf{w}_1\mathbf{x}_N + b_1) & ... & h(\mathbf{w}_L\mathbf{x}_N + b_L) \end{pmatrix}$$

On the other hand $\beta_{L\times m}$ is the matrix of output weights that contains all the $\boldsymbol{\beta}_j$ terms that connect the hidden neurons with the output neurons:

$$\beta = \begin{pmatrix} \boldsymbol{\beta}_1^T \\ \vdots \\ \boldsymbol{\beta}_L^T \end{pmatrix} = \begin{pmatrix} \beta_{1,1} & \cdots & \beta_{1,m} \\ \vdots & \ddots & \vdots \\ \beta_{L,1} & \cdots & \beta_{L,m} \end{pmatrix}$$

Finally, $T_{N\times m}$ is the target or network's output matrix:

$$T = \begin{pmatrix} \mathbf{t}_1^T \\ \vdots \\ \mathbf{t}_N^T \end{pmatrix} = \begin{pmatrix} t_{1,1} & \cdots & t_{1,m} \\ \vdots & \ddots & \vdots \\ t_{N,1} & \cdots & t_{N,m} \end{pmatrix}$$

As an SLFN, the training of the ELM is aimed to find the specific W, B and β parameters that minimize the error given by Eq. 9. In the ELM case, Eq. 9 can be represented by the following objective function:

$$\min_{W,B,\beta} \parallel H(W,B,X)\beta - T \parallel_2 \tag{12}$$

Taking into account Eqs. 11 and 12, with random W and B, the calculation of the optimal output weights matrix β can be approached as a least-squares problem in which the shortest length and minimum norm solution is given by Eq. 13:

$$\beta = H^{\dagger}T, \tag{13}$$

where $H^{\dagger}$ is the Moore–Penrose generalized inverse of the matrix H, a particular type of 1-inverse matrix. Minimizing the norm of the output weights β leads to reach a better generalization performance, resulting in a more robust solution [66, 96].

Although the ELM can approximate the output of the network to any given random set of W and B, it presents a problem with such random selection. Specifically, it makes a rather inefficient use of hidden nodes due the existence of a set of non-optimal and unnecessary input weights and hidden biases [97]. In order to mitigate this problem, an orthogonal ELM has been presented [86] to make the weights and biases orthogonal, so that $W^T \cdot W = W \cdot W^T = I$ being I the identity matrix, so $W^T = W^{-1}$ and $B^T \cdot B = B \cdot B^T = 1$. This approximation can reach a higher performance. In fact, orthogonal W and B are geometrically representing isometric transformations over Hilbert spaces, projecting $X \in \mathbb{R}^{N\times d}$ into a different or equal dimension space $\mathbb{R}^{N\times L}$ (depending on the number of hidden nodes L). This property becomes crucial to use the ELM as an unsupervised autoencoder (ELM-AE) [64–66, 82, 86], in which the desired output T is equal to the input X. In this case, the goal is to find the optimal solution to Eq. 12, modified as follows:

$$\min_{W,B,\beta} \parallel H(W,X,B)\beta - X \parallel_2 \tag{14}$$

The easy design of the basic ELM, together with its fast processing capability and easy implementation, makes it a powerful tool. However, it has been demonstrated that ELM requires more hidden neurons than other conventional tuning-based neural network algorithms, which can lead to slower performance and model overfitting. This problem becomes worse with the use of high-dimensional data such as hyperspectral remotely sensed images. To address this issue, we have developed a new fast DR technique with the goal of using ELM architectures for processing and compression of hyperspectral datasets in real-time. To achieve this, the newly developed method consists of two main parts:

1. An ELM based autoencoder, used to compress the spectral signatures (pixels) of the remotely sensed hyperspectral image.

2. A classifier that processes, analyzes and classifies these hyperspectral images.

In the following section we describe in more details these two main steps of our newly developed method.

3 Proposed method

3.1 Spectral compression

ELMs have demonstrated to be useful and efficient in compression tasks [64–66, 82, 86] with different structures. In [86], the basic ELM (see Fig. 2) is modified to perform an unsupervised learning process of a dimensionality reduction method, equaling the input to the output ($X = T$) and choosing orthogonal random W and B to project the input data space from $\mathbb{R}^{N\times d}$ to the hidden layer space $\mathbb{R}^{N\times L}$. Depending on the number of neurons in the hidden layer, L, the ELM-AE hidden layer's output can have three different behaviours: (1) if $L > d$, where d is the number of nodes in the input layer, the ELM feature function is a sparse representation of the input data X; (2) if $L = d$, the ELM feature function is an equal dimension representation of the input data X, and (3) if if $L < d$, the ELM feature function is a compressed representation of the original input X [86]. Additionally, depending on the L size, Eq. 13 can be rewritten as:

$$\beta = \left(\frac{I}{C} + H^T H\right)^{-1} H^T X \tag{15}$$

when $L > d$ or $L < d$, being $H^\dagger = \left(H^T H\right)^{-1} H^T$ the Moore-Penrose generalized inverse [98–100] and C a regularization term that makes the solution more robust and allows the ELM to achieve better generalization performance [81, 101]. On the other hand, when $L = d$, Eq. 13 can be rewritten as:

$$\beta = H^{-1}X \quad \rightarrow \quad \beta^T\beta = \beta\beta^T = I \tag{16}$$

In this case, Eq. 16 cannot been solved by the standard least squares solution given by Eq. 15 because this equation does not impose the constraint that β must be orthogonal. So, Eq. 16 must be solved as an orthogonal Procrustes problem [102], where we want to calculate the orthogonal matrix β that relates H and X. The objective function in this case will be $\min_\beta \parallel H\beta - X \parallel_F$, being $\parallel \cdot \parallel_F$ the Frobenius norm. The final solution will be $\beta = UV^T$, where U and V are the singular value decomposition (SVD) matrices of $H^T X = U\Sigma V^T$.

The architecture of our proposed fast method for hyperspectral dimension reduction has been selected as a single hidden layer ELM-AE. The model is composed by three layers: $d - L - d$ (see Fig. 2), i.e., one input layer, with d input neurons, being d the number of spectral bands; one hidden layer with L hidden neurons, being $L < d$ the compression ratio, i.e. the number of spectral bands that the hyperspectral data will preserve while the compression percentage can be calculated as $100 \cdot \left(1 - \left(\frac{L}{d}\right)\right)$%, and one output layer with d output neurons to reconstruct the input data. The decompression error has been selected as the reference measurement that will be taken into account to determine the accuracy of the method. This reference is calculated as indicated in Eq. 9 by the mean square error (MSE) as follows:

$$MSE = \frac{(X - O)^2}{N}, \tag{17}$$

where $X \in \mathbb{R}^{N\times d}$ is the desired output, $O \in \mathbb{R}^{N\times d}$ is the network output, and N is the number of samples. Once the network is trained, a test set is presented to the network. Also, the error incurred by the network during the test phase is calculated with Eq. (17).

3.2 Compressed spectrum classification

The ELM can be extended to a deeper architecture by simply stacking layer by layer to create the hierarchical structure of a multilayer ELM (ML-ELM) [86]. As we can see in Fig. 3, the ML-ELM calculates each layer's β as an independent, basic ELM, having two steps: the first random feature mapping operation, and the construction of the ML-ELM. The idea is simple; to obtain the β_i weights that relate the layer i with the layer $i - 1$, the ML-ELM constructs a classic ELM where the input layer will be layer $i - 1$ with input data $H_{i-1} = H_{i-2}\beta_{i-1}$, the hidden layer will be the layer i, with output $H_i = (W, H_{i-1}, B)$ passed thought an activation function, being W and B orthogonal random weights and biases, respectively, and the output layer will be again the layer $i - 1$. The objective function is $\min_{\beta_i} \parallel H_i\beta_i - H_{i-1} \parallel_2$ and we can employ Eq. 13 to obtain the solution $\beta_i = H_i^\dagger H_{i-1}$. The resulting β_i weights store representative information of the input data and can project the data from the defined space in layer i (data $\in \mathbb{R}^{L_i}$) into the defined space in layer $i - 1$ (data $\in \mathbb{R}^{L_{i-1}}$), so, to adapt β_i to the ML-ELM, only the transposition is needed: β_i^T.

On the other hand, to calculate each β_i, in the ML-ELM random feature mapping that involves random orthogonal weights W and biases B, the logistic sigmoid function has been used as the activation function of each auxiliary H_i:

$$H_i = \frac{1}{1 + e^{-(H_{i-1}W+B)}} \tag{18}$$

Following the same operation, a stack ELM architecture was used in order to implement our fast DR and classification method. Its architecture is composed by two main parts: (1) the compressor network, implemented as an ELM-AE, and (2) the classifier network, implemented as a classical ELM

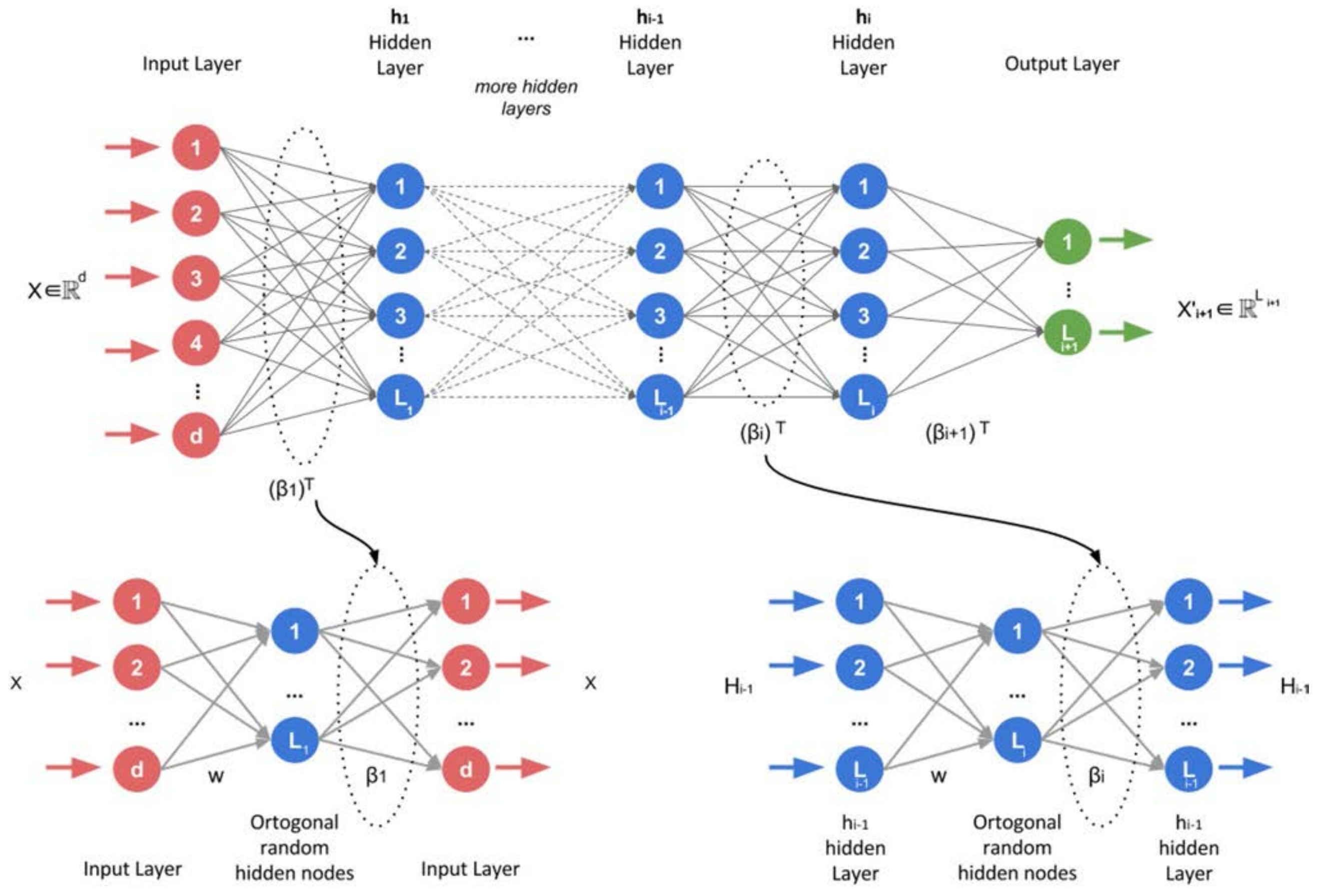


Fig. 3 ML-ELM architecture for hyperspectral image classification. To calculate each β_i of the deep network, two main steps should be performed: (1) a basic ELM random feature mapping, and (2) the transpose of the output weight β_i, to reconstruct the output ML-ELM matrix

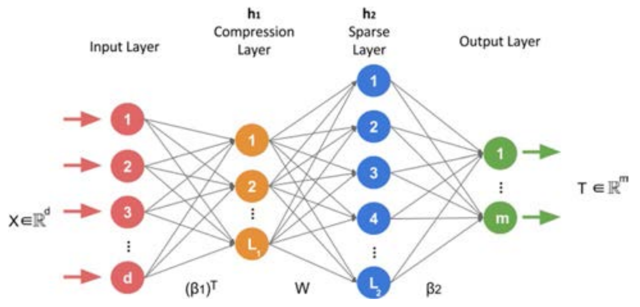
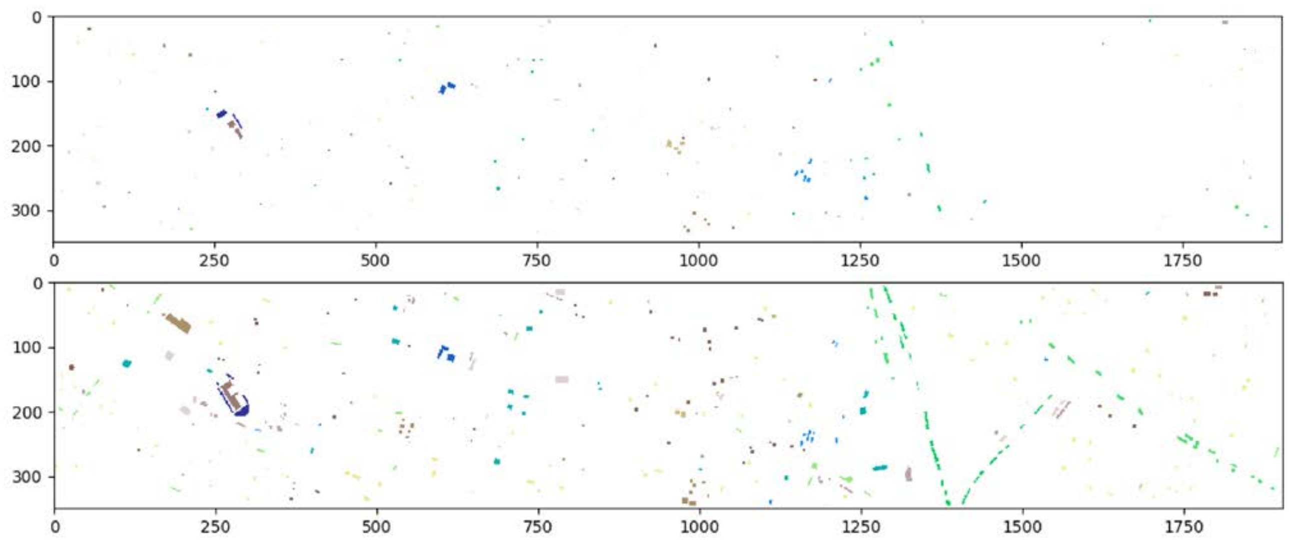


Fig. 4 Fast dimensionality reduction and classification architecture

(see Fig. 4). The network is composed by four layers with sizes $d - L_1 - L_2 - m$:

– The input layer is composed by d neurons, i.e., the spectral bands of the input data $X \in \mathbb{R}^{N\times d}$.
– The first hidden layer, or compression layer, projects the original d-dimensional data, $X \in \mathbb{R}^{N\times d}$, onto a lower dimensional subspace $\mathbb{R}^{N\times L_1}$ defined by the activations of the L_1 hidden neurons, and extracts the most representative features of the data. $L_1 < d$ indicates the compression ratio and $(\beta_1)^T$ represents the weights that connect the input layer with the compression layer, following the ML-ELM procedure.
– The second hidden layer, or sparse layer, expands the data representation, cleaning the data for later classification. It is composed by $L_2 > L_1$ neurons, being W the random orthogonal weights that connect the output of the compression layer with the sparse layer.
– Finally, the output layer performs the classification task and is composed by m neurons, being m the number of classes and β_2 the ELM weights that connect the sparse layer with the output.

These four layers are trained separately, being the ELM-AE trained with a different training percentage than the ELM-classifier. In the test phase, both networks are stacked to obtain the classification results with compressed data.



Color	Land cover type	Samples train	Samples test	Color	Land cover type	Samples train	Samples test
	Background	649816		⬜	Grass-healthy	198	1053
🟫	Grass-stressed	190	1064	🟫	Grass-synthetic	192	505
🟫	Tree	188	1056	🟫	Soil	186	1056
🟨	Water	182	143	🟨	Residential	196	1072
🟩	Commercial	191	1053	🟩	Road	193	1059
🟩	Highway	191	1036	🟩	Railway	181	1054
🟦	Parking-lot1	192	1041	🟦	Parking-lot2	184	285
🟦	Tennis-court	181	247	🟦	Running-track	187	473
			Total samples 2832|12197				

Fig. 5 Ground-truth of the Houston 349 × 1905 × 144 hyperspectral scene: Training data (top) with a total of 2832 samples and testing data (bottom) with a total of 12197 samples

4 Experimental validation

4.1 Experimental configuration

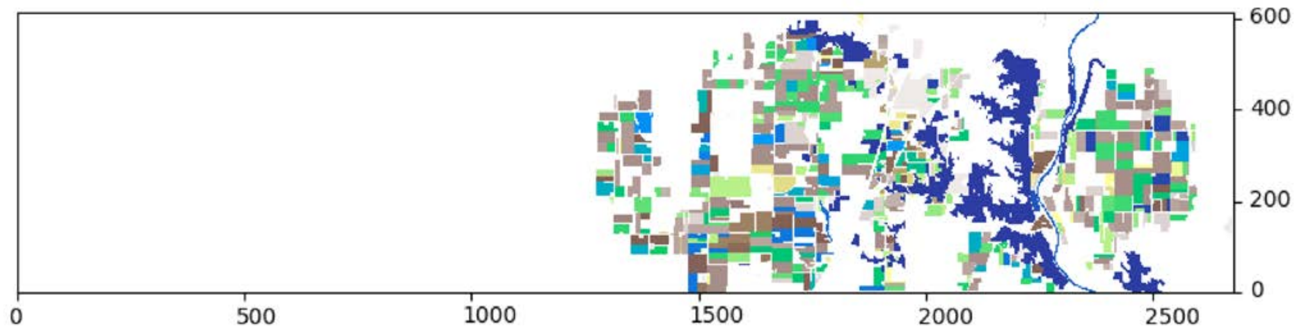
To test the performance of our proposed fast DR and classification method for hyperspectral remote sensing image compression, a comparison between the proposed method and a PCA compressor and MLP-based autoencoder has been carried out. The hardware architecture used for the comparison is composed of a 6th Generation Intel® Core™i7-6700K processor with 8M of Cache and up to 4.20 GHz (4 cores/8 way multitask processing), 64GB of DDR4 RAM with a serial speed of 2400 MHz, a Toshiba DT01ACA HDD with 7200 RPM and 2TB of capacity, and an ASUS Z170 pro-gaming motherboard.

On the other part, the software environment is composed by Ubuntu 16.04 as operating system. Proposed method has been implemented on python, using the libraries for scientific computing Scipy and Numpy. Additionally, to compare with a GPU-implemented multilayer perceptron (MLP), CUDA 8 has been employed.

4.2 Hyperspectral datasets

Compression (and posterior classification) experiments have been carried out using five hyperspectral remote sensing images, characterized by the availability of categorized ground-truth with the objective of testing the method in several applications, in particular to analyze the effect of compression on classification, with different spectral-spatial characteristics:

1. The first hyperspectral scene is known as Indian Pines. It was collected by he Airborne Visible-Infrared Imaging Spectrometer (AVIRIS) [103] in 1992 over a set of agricultural fields with regular geometry, with multiple crops and irregular forest areas in northwestern Indiana. This scene forms a data cube of dimensions 145 × 145 × 224, with a spatial resolution of 20m per pixel. The spectral bands capture the solar spectrum in the range from 0.4 to 2.5 µm, being 4 zero bands and another 20 bands with lower SNR because of atmospheric absorption bands that are removed, preserving

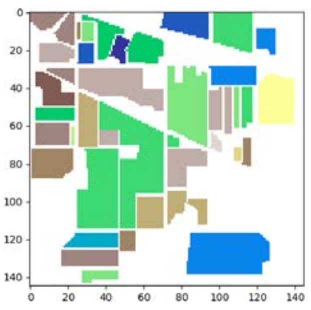


Color	Land cover type	Samples	Color	Land cover type	Samples
	Background	1310047	⬜	BareSoil	57
⬜	Buildings	17195	⬜	Concrete/Asphalt	69
⬜	Corn	17783	⬜	Corn?	158
🟫	Corn-EW	514	🟫	Corn-NS	2356
🟫	Corn-CleanTill	12404	🟫	Corn-CleanTill-EW	26486
🟫	Corn-CleanTill-NS	39678	🟫	Corn-CleanTill-NS-Irrigated	800
🟫	Corn-CleanTilled-NS?	1728	🟫	Corn-MinTill	1049
🟫	Corn-MinTill-EW	5629	🟫	Corn-MinTill-NS	8862
🟫	Corn-NoTill	4381	🟫	Corn-NoTill-EW	1206
🟫	Corn-NoTill-NS	5685	🟫	Fescue	114
🟨	Grass	1147	🟨	Grass/Trees	2331
🟨	Grass/Pasture-mowed	19	🟨	Grass/Pasture	73
🟨	Grass-runway	37	🟨	Hay	1128
🟨	Hay?	2185	🟨	Hay-Alfalfa	2258
🟨	Lake	224	🟨	NotCropped	1940
🟨	Oats	1742	🟩	Oats?	335
🟩	Orchard	39	🟩	Pasture	10386
🟩	pond	102	🟩	Soybeans	9391
🟩	Soybeans?	894	🟩	Soybeans-NS	1110
🟩	Soybeans-CleanTill	5074	🟩	Soybeans-CleanTill?	2726
🟩	Soybeans-CleanTill-EW	11802	🟩	Soybeans-CleanTill-NS	10387
🟩	Soybeans-CleanTill-Drilled	2242	🟩	Soybeans-CleanTill-Weedy	543
🟩	Soybeans-Drilled	15118	🟩	Soybeans-MinTill	2667
🟦	Soybeans-MinTill-EW	1832	🟦	Soybeans-MinTill-Drilled	8098
🟦	Soybeans-MinTill-NS	4953	🟦	Soybeans-NoTill	2157
🟦	Soybeans-NoTill-EW	2533	🟦	Soybeans-NoTill-NS	929
🟦	Soybeans-NoTill-Drilled	8731	🟦	Swampy Area	583
🟦	River	3110	🟦	Trees?	580
🟦	Wheat	4979	🟦	Woods	63562
🟪	Woods?	144			
		Total samples 1644292			

Fig. 6 Ground-truth of the large Indian Pines 2678 × 614 × 220 hyperspectral scene. Labels with "?" correspond with categories related to the name referred but which have been defined as independent classes

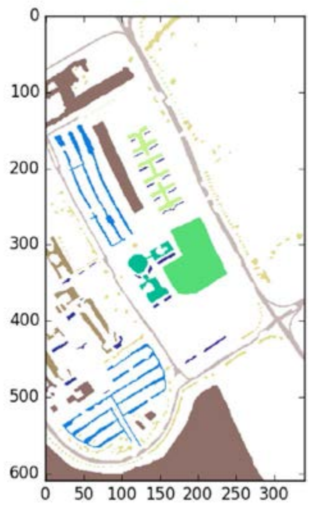
the remaining 200 channels. About half of the pixels in the image (10366 of 21025) contain ground-truth information, which comes in the form of a single label assignment having a total of 16 ground-truth classes (see Fig. 7).

2. The second hyperspectral scene is a large version of Indian Pines called large Indian Pines. Also, it was collected by AVIRIS [103] in 1992 over a set of agricultural fields and irregular patches of forest in Northwestern Indiana over the same area, but spanning a much larger



Color	Land cover type	Samples
	Background	10776
■	Alfalfa	46
■	Corn-notill	1428
■	Corn-min	830
■	Corn	237
■	Grass/Pasture	483
■	Grass/Trees	730
■	Grass/pasture-mowed	28
■	Hay-windrowed	478
■	Oats	20
■	Soybeans-notill	972
■	Soybeans-min	2455
■	Soybean-clean	593
■	Wheat	205
■	Woods	1265
■	Bldg-Grass-Tree-Drives	386
■	Stone-steel towers	93
	Total samples	21025

Fig. 7 Ground-truth of the Indian Pines 145 × 145 × 200 hyperspectral scene

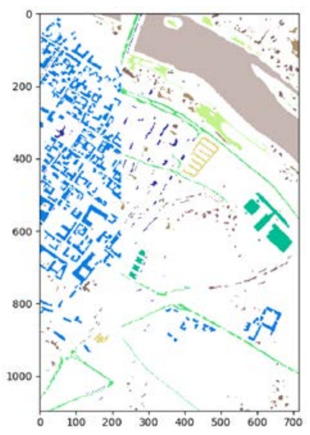


Color	Land cover type	Samples
	Background	164624
■	Asphalt	6631
■	Meadows	18649
■	Gravel	2099
■	Trees	3064
■	Painted metal sheets	1345
■	Bare Soil	5029
■	Bitumen	1330
■	Self-Blocking Bricks	3682
■	Shadows	947
	Total samples	207400

Fig. 8 Ground-truth of the Pavia University 610 × 340 × 103 hyperspectral scene

extent, with a size of 2678 × 614 pixels containing 220 spectral bands with 58 ground-truth classes (see Fig. 6).

3. The third hyperspectral scene is known as Pavia University. It was collected by the Reflective Optics System Imaging Spectrometer (ROSIS) sensor [104] during a flight campaign in northern Italy, over the city of Pavia, covering an urban environment, with solid structures, natural objects and their shadows. Fig. 8 shows this scene comprising 9 classes. It contains 103 spectral bands of 610 × 340 pixels in the spectral range from 0.43 to 0.86 μm, with spatial resolution of 1.3 m/pixel.
4. The fourth hyperspectral dataset is known as Pavia Centre (Fig. 9), and it was collected by ROSIS too, during a flight campaign over Pavia. This scene contains 1096 × 715 pixels with 102 spectral bands, with geometric resolution of 1.3 meters. The ground truth contains 9 different classes.
5. Finally, the fifth hyperspectral dataset is known as Houston [105]. It was collected by the Compact Airborne Spectrographic Imager (CASI) in June 2012 over the University of Houston campus and the neighboring urban area. This scene forms a cube of dimension 349 × 1905 × 144, with spatial resolution of 2.5 m and spectral information captured in the range from 0.38 to 1.05 μm, containing 15 ground-truth classes (see Fig. 5) divided in two categories: training and testing.



Color	Land cover type	Samples
	Background	635488
■	Water	65971
■	Trees	7598
■	Asphalt	3090
■	Self-Blocking Bricks	2685
■	Bitumen	6584
■	Tiles	9248
■	Shadows	7287
■	Meadows	42826
■	Bare Soil	2863
	Total samples	783640

Fig. 9 Ground-truth of the Pavia Centre 1096 × 715 × 102 hyperspectral scene

4.3 Performance evaluation

To validate the proposed implementation, two kinds of experiments have been conducted: the first one has been designed to check the process of compression and decompression, and the second one has been designed to evaluate the classification results for the images compressed by the proposed fast method. In both cases, $X \in \mathbb{R}^{N\times d}$, random samples are acquired from the original input data and split into a training and testing set. For the compression experiment, 85% of X has been selected for the training phase, using the remaining 15% of X for testing. For compression and classification, 1, 2, 3, 5, and 10% of the compressed X has been used for the training phase and the remaining 99, 98, 97, 95, and 90% for the testing phase. In both cases, the data are pre-processed with L_2 normalization, to prevent the premature saturation of the mapping nodes. Additionally, has been executed ten times and the average values are reported, together with the standard deviation, to measure the robustness of the method.

Table 1 Execution times and decompression MSE for the small Indian Pines image compression experiment

Components	Ratio (%)	PCA	ICA	NMF	MLP-AE	Proposed
Time execution						
10	95	3.66e−02 (2.90e−03)	9.37e−01 (7.81e−02)	3.21e+00 (1.14e−02)	6.96e+00 (2.37e−01)	**3.07e−02** (1.03e−03)
30	85	**3.42e−02** (1.26e−04)	4.81e+00 (5.58e−01)	7.29e+00 (2.95e−02)	7.11e+00 (1.01e−01)	5.84e−02 (7.73e−05)
50	75	**3.43e−02** (1.97e−04)	7.85e+00 (2.81e−02)	2.11e+01 (1.23e−01)	7.37e+00 (9.51e−02)	8.37e−02 (3.59e−04)
70	65	**3.49e−02** (1.47e−03)	1.09e+01 (2.63e−02)	3.25e+01 (7.50e−02)	7.60e+00 (7.88e−02)	1.19e−01 (1.59e−03)
90	55	**3.44e−02** (2.50e−04)	1.41e+01 (3.78e−02)	4.96e+01 (4.32e−01)	7.79e+00 (5.54e−02)	1.78e−01 (6.58e−04)
110	45	**3.46e−02** (1.18e−04)	1.75e+01 (5.64e−02)	7.16e+01 (2.54e−01)	8.01e+00 (7.08e−02)	2.64e−01 (6.17e−03)
130	35	**3.45e−02** (1.24e−04)	1.93e+01 (3.94e−02)	9.40e+01 (2.77e+00)	8.28e+00 (1.07e−01)	2.40e−01 (8.24e−03)
150	25	**3.45e−02** (1.17e−04)	2.23e+01 (2.45e−02)	1.23e+02 (2.11e+00)	8.53e+00 (8.03e−02)	3.41e−01 (2.02e−03)
170	15	**3.44e−02** (8.31e−05)	2.53e+01 (5.56e−02)	1.56e+02 (5.61e+00)	8.74e+00 (1.36e−01)	3.36e−01 (1.70e−03)
190	5	**3.44e−02** (1.26e−04)	2.83e+01 (3.44e−02)	1.90e+02 (5.47e+00)	8.96e+00 (9.02e−02)	4.45e−01 (2.49e−03)
Mean square error						
10	95	1.78e−02 (5.73e−05)	1.78e−02 (5.62e−05)	**2.84e−07** (2.26e−09)	6.65e−06 (1.48e−06)	7.41e−07 (1.25e−07)
30	85	3.89e−03 (1.86e−05)	3.90e−03 (1.58e−05)	**1.12e−07** (7.06e−10)	5.22e−06 (4.49e−07)	2.15e−07 (9.35e−09)
50	75	1.96e−03 (5.79e−06)	1.96e−03 (3.43e−06)	**6.27e−08** (5.79e−10)	4.42e−06 (2.02e−07)	1.19e−07 (2.55e−09)
70	65	1.23e−03 (6.69e−06)	1.23e−03 (1.19e−06)	**4.36e−08** (8.08e−10)	4.19e−06 (2.55e−07)	7.40e−08 (1.32e−09)
90	55	7.59e−04 (5.91e−06)	7.53e−04 (1.39e−06)	**3.12e−08** (6.58e−10)	4.06e−06 (1.48e−07)	4.75e−08 (7.34e−10)
110	45	4.78e−04 (4.53e−06)	4.71e−04 (1.56e−06)	**2.03e−08** (5.93e−10)	4.08e−06 (1.93e−07)	3.07e−08 (5.89e−10)
130	35	2.92e−04 (4.83e−06)	2.85e−04 (7.43e−07)	**1.22e−08** (4.30e−10)	4.17e−06 (2.32e−07)	1.93e−08 (4.47e−10)
150	25	1.59e−04 (5.02e−06)	1.51e−04 (2.93e−07)	**9.10e−09** (3.56e−10)	4.57e−06 (3.76e−07)	1.23e−08 (2.72e−10)
170	15	6.92e−05 (5.19e−06)	6.20e−05 (1.04e−07)	7.61e−09 (1.30e−10)	4.42e−06 (3.88e−07)	**7.41e−09** (2.59e−10)
190	5	2.37e−05 (5.21e−06)	1.65e−05 (6.08e−08)	6.18e−09 (1.22e−10)	5.23e−06 (7.42e−07)	**3.99e−09** (1.34e−10)

Best values obtained are given in bold

4.3.1 Testing compression–decompression performance

In order to test the compression–decompression functionality of our method, this work makes a comparison between a single hidden layer ELM based implementation of our method with a multicore PCA, the parallel FastICA, the NMF algorithm and GPU MLP-based autoencoder, taking into account the decompression error of each algorithm. The characteristics of our single hidden layer ELM based approach have been described in Sect. 3.1, with an architecture $d - L - d$, being d the number of spectral bands and L the compression ratio. The topology of the MLP based autoencoder has been chosen following the same single layer ELM architecture, implementing sigmoid function as activation function in all its layers and Adam [106] as optimizer with learning rate set to 0.001, batch size of 512 and 100 epochs. All compression methods have been trained over the 85% of the data, being the remaining 15% used for testing.

Table 1 shows the execution times and decompression error of each method over small Indian Pines scene. In this case, PCA, ICA, NMF, MLP-AE and proposed method have used 10 different compression ratio, defined by L. In fact, they have compressed the 95, 85, 75, 65, 55, 45, 35, 25, 15, 5% of the spectral bands, setting L to 10, 30, 50, 70, 90, 110, 130, 150, 170 and 190, respectively. In general, we can observe in the five methods that the execution time increases as L increases, while the reconstruction error is more independent. In particular, PCA is the fastest DR method due to the simplicity of its algorithm (reduced to multiplications of matrices), followed closely by proposed DR method, while NMF is the slowest one, followed by ICA and MLP-AE. Comparing execution times of ELM-AE, NMF, ICA and MLP-AE we can observe a speedup of 357.05, 72.19 and 37.86 on average, respectively, while PCA has an average speedup over proposed DR method of 6.04. On the other part, although PCA provides the lowest execution times, it also provides the worst reconstruction error together with ICA algorithm. On the other part, NMF provides the best MSE values, followed very closely by the proposed DR method (with a distance of 6.81e−08 on average), which is able to reach better MSE values than PCA, ICA an MLP-AE and provides the lowest MSE value when compressing the 15% of spectral bands, setting L to 170 bands. The aforementioned behaviour is consistent in the experiments with all the considered hyperspectral scenes, being the single layer ELM the one with the best MSE-execution time tradeoff, with MSE better than PCA and similar or even better than MLP and NMF methods (but much faster performance).

Fig. 10 shows the reconstruction of a randomly selected spectral signature of the small Indian Pines using the

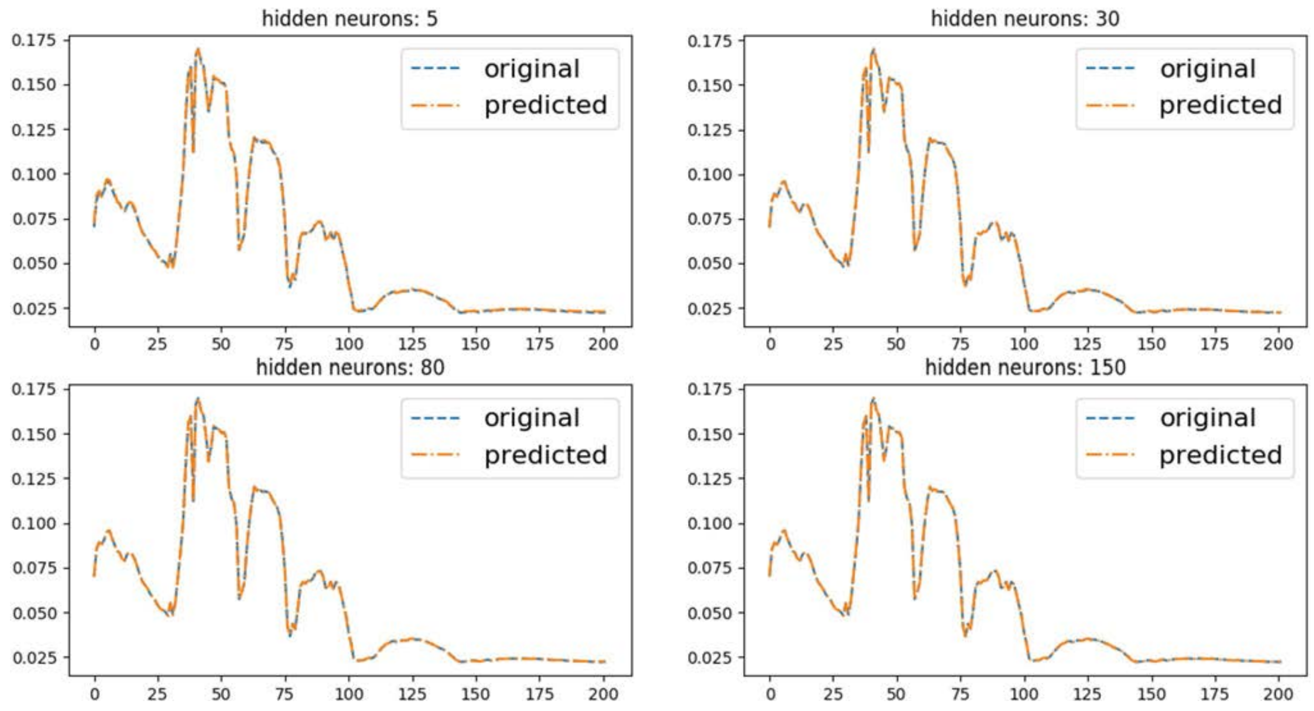


Fig. 10 Fast spectral reconstruction of a randomly-selected spectral signature from small Indian Pines scene with compression ratio $L = 5$ (97.5%), 30 (85%), 80 (60%) and 150 (25%)

proposed method as compressor. We can see that both the original signature (blue) and the reconstructed signature (orange) are very similar, even when compression percentages are high, such as compressing the 97.5% of the spectral bands ($L = 5$). Additionally, in Fig. 11, we can see the reconstruction error of the proposed method. Each row is composed by the false color map of band b, from the original data cube (before compressing the 80% of spectral bands with $L = 40$, left), the obtained band b after decompression (center) and the difference (MSE) between the original spectral value and the decompressed or reconstructed value obtained by the proposed DR method (right). Color-bar indicates that differences that are closely to zero corresponds with white color, while dark-blue color indicates the maximum difference between the original spectral band and the reconstructed one. As we can observe, both maps (original and reconstructed) are quite similar.

Table 2 shows the execution times and the decompression error using the Pavia University hyperspectral scene, reached by each DR method: PCA, ICA, NMF, MLP-based autoencoder and the proposed method based on a single layer ELM. In this case, 5 different compression ratios have been used, determined by the size of L, i.e., the number of output spectral bands in compressed image, in particular 10 (we round the percentage for better understanding, compressing about the 90.29–90% of bands), 30 (70.87–71%), 50 (51.46–51%), 70 (32.04–32%) and 90 (12.62–13%). In this hyperspectral scene, Pavia University has 207 400 pixels, 9.86 times greater than small Indian Pines, which has 186 375 pixels less, so the execution times are increased in all the studied DR methods. Again, PCA becomes the faster one, with a speedup of 10.22 over ELM-AE on average. NMF, MLP-AE and ICA are the slowest again, having the proposed method a speedup of 160.41, 39.71 and 32.79 over them on average. Also, the PCA reaches the poorest reconstruction error, being NMF the best compression method when performs the compression of the 71, 51, 32, and 13% of the spectral bands, setting $L = 30$, 50, 70 and 90, respectively. These results are followed very closely by MLP-based autoencoder (whose MSE differs from the NMF one by 1.70e−06 points on average) and by the proposed single layer ELM (in fact, MSEs of NMF and ELM differ on 3.83e−06 points on average), with an acceptable execution time (the slowest execution has only 2.85 s).

Table 3 shows the execution times and decompression error of each considered method using the Pavia Centre hyperspectral scene. Again, 5 different compression ratios have been selected, fixing L to 10 (about the 90.20–90% of the spectral bands has been compressed), 30 (70.59–71%), 50 (50.98-51%), 70 (31.37–31%) and 90 (11.76–12%). In this case, Pavia Centre has 783 640 pixels, that means it is 3.78 times larger than Pavia University, which has 576 240 pixels less, and 37.27 times larger than small Indian Pines. As in the previous experiments the increase in the number of pixels

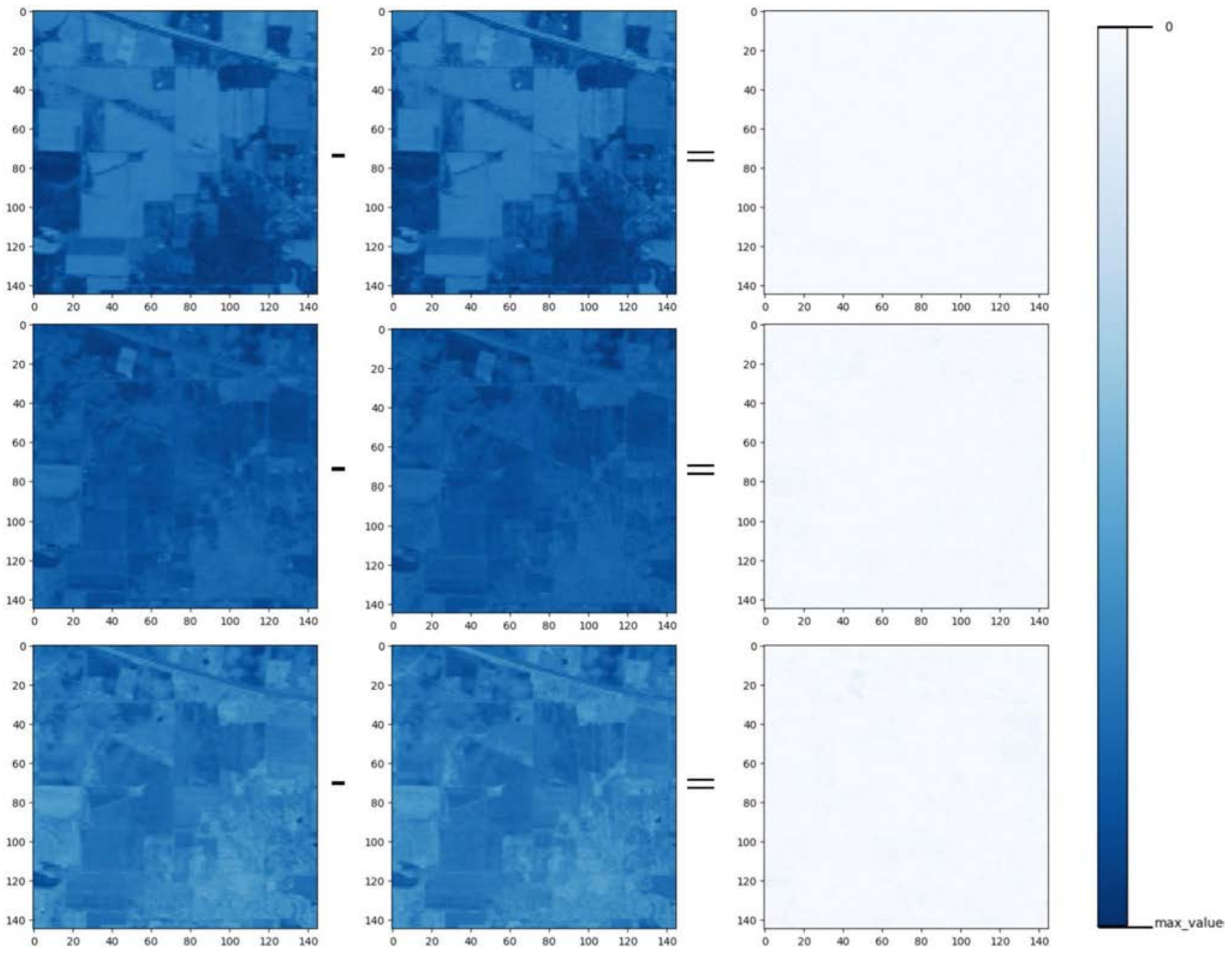


Fig. 11 Fast reconstruction with $L = 40$ (80%). Comparison between the original false-color band (left), the decompressed band with the proposed fast method (center), and the difference between them (right). This comparison has been made using spectral bands 50, 100 and 150

Table 2 Execution times and decompression MSE of the Pavia University compression experiment

Components	Ratio (%)	PCA	ICA	NMF	MLP-AE	Proposed
Time execution						
10	90	**1.31e−01** (1.22e−03)	8.73e+00 (3.51e−01)	2.10e+01 (6.69e−02)	5.11e+01 (3.09e−01)	1.73e−01 (3.25e−03)
30	71	**1.31e−01** (2.07e−04)	1.85e+01 (2.42e+00)	7.83e+01 (8.26e−02)	5.20e+01 (2.83e−01)	5.78e−01 (1.63e−02)
50	51	**1.31e−01** (1.32e−04)	5.87e+01 (1.29e+01)	1.75e+02 (5.32e−01)	5.31e+01 (2.84e−01)	1.17e+00 (3.30e−02)
70	32	**1.31e−01** (1.65e−04)	5.54e+01 (1.60e+01)	2.98e+02 (1.07e+00)	5.42e+01 (4.70e−01)	1.92e+00 (4.98e−02)
90	13	**1.31e−01** (1.67e−04)	7.81e+01 (1.60e+01)	5.01e+02 (6.33e+01)	5.53e+01 (4.10e−01)	2.85e+00 (5.74e−02)
Mean square error						
10	90	2.09e−03 (1.90e−04)	2.12e−03 (1.71e−04)	6.08e−06 (9.36e−08)	**5.95e−06** (9.94e−08)	2.20e−05 (5.29e−06)
30	71	4.18e−04 (5.55e−05)	4.29e−04 (4.67e−05)	**1.52e−06** (2.78e−08)	2.97e−06 (1.24e−07)	3.58e−06 (1.20e−07)
50	51	1.17e−04 (1.04e−05)	1.17e−04 (6.83e−06)	**6.47e−07** (1.69e−08)	2.65e−06 (1.37e−07)	1.43e−06 (6.39e−08)
70	32	2.68e−05 (3.52e−06)	2.63e−05 (2.00e−06)	**2.81e−07** (1.29e−08)	2.69e−06 (2.30e−07)	5.89e−07 (2.74e−08)
90	13	4.38e−06 (1.02e−06)	3.78e−06 (6.85e−07)	**1.41e−07** (7.40e−09)	2.89e−06 (2.03e−07)	2.39e−07 (2.19e−08)

Best values obtained are given in bold

Table 3 Execution times and decompression MSE of the Pavia Centre compression experiment

Components	Ratio (%)	PCA	ICA	NMF	MLP-AE	Proposed
Time execution						
10	90	**4.78e−01** (9.60e−03)	3.44e+01 (1.90e+00)	1.09e+02 (6.06e+00)	2.12e+02 (1.08e+00)	7.26e−01 (6.59e−03)
30	71	**4.73e−01** (7.65e−04)	6.72e+01 (3.90e+00)	3.01e+02 (8.25e−01)	2.15e+02 (1.68e+00)	2.32e+00 (3.35e−02)
50	51	**4.75e−01** (7.16e−03)	1.47e+02 (1.58e+01)	6.66e+02 (6.36e+00)	2.20e+02 (1.18e+00)	4.66e+00 (4.91e−02)
70	31	**4.73e−01** (7.06e−04)	2.11e+02 (1.12e+01)	1.13e+03 (3.17e+00)	2.23e+02 (1.80e+00)	7.45e+00 (4.18e−02)
90	12	**4.73e−01** (5.25e−04)	2.63e+02 (4.18e+01)	1.73e+03 (5.15e+00)	2.29e+02 (4.08e+00)	1.11e+01 (1.19e−01)
Mean square error						
10	90	2.26e−03 (1.30e−04)	2.27e−03 (1.48e−04)	1.97e−05 (4.75e−07)	**1.78e−05** (2.26e−07)	6.03e−05 (7.79e−06)
30	71	3.54e−04 (2.86e−05)	3.61e−04 (3.33e−05)	5.54e−06 (1.14e−07)	**4.59e−06** (1.10e−07)	1.27e−05 (8.38e−07)
50	51	9.84e−05 (9.98e−06)	1.01e−04 (1.24e−05)	2.35e−06 (7.59e−08)	**1.98e−06** (6.28e−08)	5.21e−06 (2.15e−07)
70	31	2.21e−05 (2.67e−06)	2.24e−05 (3.15e−06)	**1.01e−06** (2.75e−08)	1.42e−06 (2.24e−07)	2.24e−06 (1.12e−07)
90	12	3.08e−06 (5.89e−07)	2.93e−06 (5.93e−07)	**4.47e−07** (2.12e−08)	1.52e−06 (4.62e−07)	1.04e−06 (4.33e−08)

Best values obtained are given in bold

Table 4 Execution times and decompression MSE of the Houston compression experiment

Components	Ratio (%)	PCA	ICA	NMF	MLP-AE	Proposed
Time execution						
10	93	**5.08e−01** (8.78e−04)	4.15e+01 (1.29e+00)	8.97e+01 (7.39e−02)	1.84e+02 (1.66e+00)	6.87e−01 (5.36e−03)
30	79	**5.10e−01** (2.07e−03)	7.34e+01 (8.41e+00)	2.79e+02 (6.62e−01)	1.87e+02 (9.00e−01)	2.06e+00 (3.43e−02)
50	65	**5.11e−01** (6.56e−03)	1.31e+02 (3.97e+01)	6.06e+02 (1.51e+00)	1.91e+02 (1.09e+00)	4.08e+00 (7.10e−02)
70	51	**5.14e−01** (9.58e−03)	1.71e+02 (3.37e+01)	1.02e+03 (1.74e+00)	1.94e+02 (1.15e+00)	6.52e+00 (1.10e−01)
90	38	**5.09e−01** (1.79e−03)	3.13e+02 (5.12e+01)	1.52e+03 (2.89e+00)	1.99e+02 (3.00e+00)	9.62e+00 (2.84e−01)
110	24	**5.13e−01** (8.16e−03)	4.03e+02 (4.43e+01)	2.32e+03 (7.30e+00)	2.02e+02 (1.30e+00)	1.30e+01 (3.52e−01)
130	10	**5.12e−01** (6.34e−03)	4.87e+02 (1.39e+00)	3.14e+03 (1.10e+01)	2.05e+02 (1.22e+00)	1.30e+01 (1.85e−01)
Mean square error						
10	93	1.11e−03 (1.72e−04)	1.14e−03 (2.03e−04)	9.33e−07 (1.74e−08)	**8.51e−07** (4.96e−08)	3.61e−06 (6.57e−07)
30	79	3.73e−04 (8.65e−05)	3.84e−04 (9.49e−05)	**3.45e−07** (1.17e−08)	7.50e−07 (1.87e−07)	6.84e−07 (3.59e−08)
50	65	2.27e−04 (7.35e−05)	2.44e−04 (7.98e−05)	**2.25e−07** (8.87e−09)	7.37e−07 (7.00e−08)	3.90e−07 (1.54e−08)
70	51	1.29e−04 (5.56e−05)	1.30e−04 (6.22e−05)	**1.75e−07** (7.04e−09)	7.64e−07 (1.30e−07)	2.57e−07 (1.22e−08)
90	38	7.13e−05 (3.44e−05)	7.04e−05 (3.89e−05)	**1.36e−07** (4.98e−09)	7.54e−07 (1.03e−07)	1.76e−07 (7.56e−09)
110	24	3.92e−05 (2.58e−05)	4.79e−05 (3.07e−05)	**1.04e−07** (5.15e−09)	7.59e−07 (7.70e−08)	1.15e−07 (4.06e−09)
130	10	2.05e−05 (1.61e−05)	2.67e−05 (1.77e−05)	7.82e−08 (3.25e−09)	7.49e−07 (9.64e−08)	**6.89e−08** (3.36e−09)

Best values obtained are given in bold

results in an increase in the computation time of all compression methods, being PCA the fastest one, whose average speedup over ELM-AE is 11.07. Also, NMF, MLP-AE and ICA are the slowest methods, while the ELM-AE's average speedup over NMF is 149.91, over MLP-AE is 41.86 and over ICA is 27.52, being the slowest execution of the single layer ELM only 11.1 s when the 12% of the spectral bands are compressed with $L = 90$, and the fastest one 0.73 second with $L = 10$ (90%). Pavia Center's spectral characteristics allow the MLP-based autoencoder to reach the best decompression errors when the 90% ($L = 10$), 71% ($L = 30$) and 51% ($L = 50$) of spectral bands are compressed, while NMF reaches the best MSE values when the 31% ($L = 70$) and 12% ($L = 90$) of spectral bands are compressed, although the single layer ELM can reach MSE values similar to MLP and NMF and NMF (the average MSE difference between MLP-AE and ELM-AE is 1.08e−05, while the difference between NMF and ELM-AE is 1.05e−05) in a much shorter time. With the Pavia University and Pavia Centre datasets we have shown that, despite of being slower than the PCA (and with results slightly inferior than the NMF and MLP-based autoencoder), our single layer ELM reaches a good balance between computation time and compression result, being a good option to compress hyperspectral images.

Another large hyperspectral scene is the Houston data set, with 664 845 pixels, i.e., 31.62 times larger than the

Table 5 Execution times and decompression MSE of the large Indian Pines compression experiment

Components	Ratio (%)	PCA	ICA	NMF	MLP-AE	Proposed
Time execution						
10	95	**2.10e+00** (8.41e−03)	1.49e+02 (2.19e+00)	3.03e+02 (7.72e−01)	5.21e+02 (1.44e+01)	2.07e+00 (8.07e−02)
30	86	**2.09e+00** (5.27e−03)	2.38e+02 (9.28e+01)	7.98e+02 (1.10e+00)	5.33e+02 (4.15e+00)	5.70e+00 (1.43e−01)
50	77	**2.09e+00** (5.84e−03)	3.72e+02 (1.47e+01)	1.60e+03 (1.77e+00)	5.33e+02 (5.10e+00)	1.07e+01 (1.83e−01)
70	68	**2.09e+00** (6.09e−03)	7.43e+02 (1.57e+01)	2.59e+03 (2.81e+01)	5.34e+02 (8.79e+00)	1.68e+01 (2.36e−01)
90	59	**2.09e+00** (6.13e−03)	6.52e+02 (1.59e+02)	3.81e+03 (9.57e+00)	5.52e+02 (8.21e+00)	2.46e+01 (6.53e−01)
110	50	**2.09e+00** (5.83e−03)	1.00e+03 (2.10e+02)	5.76e+03 (1.19e+01)	5.62e+02 (4.83e+00)	3.39e+01 (8.97e−01)
130	41	**2.09e+00** (6.83e−03)	1.11e+03 (2.48e+02)	–	5.69e+02 (3.48e+00)	3.36e+01 (8.86e−01)
150	32	**2.09e+00** (6.71e−03)	1.39e+03 (2.27e+02)	–	5.73e+02 (4.13e+00)	4.30e+01 (1.68e+00)
170	23	**2.09e+00** (6.41e−03)	1.74e+03 (1.11e+02)	–	5.79e+02 (4.94e+00)	4.33e+01 (1.54e+00)
190	14	**2.10e+00** (6.48e−03)	1.39e+03 (2.13e+02)	–	6.00e+02 (5.10e+00)	5.39e+01 (1.61e+00)
210	5	**2.10e+00** (5.36e−03)	1.68e+03 (1.40e+02)	–	6.19e+02 (1.15e+01)	5.43e+01 (1.47e+00)
Mean square error						
10	95	8.28e−02 (6.16e−04)	8.28e−02 (7.11e−04)	3.40e−07 (5.01e−09)	**3.36e−07** (1.09e−08)	9.02e−07 (7.25e−08)
30	86	1.40e−02 (8.76e−05)	1.40e−02 (6.51e−05)	**1.35e−07** (6.04e−10)	2.10e−07 (1.16e−08)	2.59e−07 (1.03e−08)
50	77	3.60e−03 (6.97e−06)	3.61e−03 (5.22e−06)	**7.64e−08** (3.45e−10)	1.91e−07 (9.56e−09)	1.45e−07 (3.79e−09)
70	68	1.78e−03 (4.33e−06)	1.78e−03 (4.51e−06)	**5.02e−08** (3.54e−10)	1.86e−07 (7.73e−09)	9.41e−08 (1.27e−09)
90	59	1.12e−03 (3.84e−06)	1.13e−03 (3.71e−06)	**3.51e−08** (1.69e−10)	1.82e−07 (5.61e−09)	6.30e−08 (9.14e−10)
110	50	7.42e−04 (2.83e−06)	7.44e−04 (2.79e−06)	**2.36e−08** (2.17e−10)	1.78e−07 (5.33e−09)	4.30e−08 (8.89e−10)
130	41	4.69e−04 (1.52e−06)	4.69e−04 (1.13e−06)	–	1.80e−07 (7.67e−09)	**2.95e−08** (4.43e−10)
150	32	2.75e−04 (9.01e−07)	2.76e−04 (8.49e−07)	–	1.81e−07 (6.32e−09)	**1.99e−08** (2.27e−10)
170	23	1.52e−04 (5.53e−07)	1.53e−04 (5.98e−07)	–	1.80e−07 (8.05e−09)	**1.32e−08** (2.71e−10)
190	14	7.42e−05 (2.05e−07)	7.42e−05 (2.00e−07)	–	1.89e−07 (1.80e−08)	**8.17e−09** (1.62e−10)
210	5	2.05e−05 (9.64e−08)	2.04e−05 (8.06e−09)	–	1.50e−07 (1.72e−08)	**4.50e−09** (1.72e−10)

Best values obtained are given in bold

small Indian Pines, 3.21 times larger than Pavia University and 1.18 times smaller than Pavia Centre. Table 4 shows the execution times and the decompression error reached by each compression method: PCA, ICA, NMF, MLP-based autoencoder and the single layer ELM-AE, with 7 different compression ratios, reducing about the 93.06–93% of the spectral bands with $L = 10$, the 79.17–79% with $L = 30$, the 65.28–65% with $L = 50$, the 51.39–51% with $L = 70$, the 37.5–38% with $L = 90$, the 23.61–24% with $L = 110$ and the 9.72–10% with $L = 130$. In this case, although PCA is the faster method with 13.68 as average speedup over ELM-AE method, it provides the worst MSE values. The proposed method exhibits good decompression MSE results in less time than NMF and MLP-AE algorithm, with 183.28 and 27.81 of average speedup, respectively (33.08 of speedup compared with ICA). Once more, NMF, ICA and MLP-based autoencoder are the slowest methods, being the MSE results of NMF and MLP-AE very similar to the single layer ELM (the MSE difference between ELM-AE and NMF is 4.72e−07, while between ELM-AE and MLP-AE is 9.01e−09). The single layer ELM offers again a good balance between computation time and compression results.

Finally, we also tested the proposed DR method with a larger hyperspectral scene, such as the large Indian Pines (with 1 644 292 pixels, i.e., 78.21 times larger than small Indian Pines, 79.28 times larger than Pavia University, 20.98 times larger than Pavia Centre and 24.73 times larger than Houston scene). Table 5 shows the obtained results in terms of execution time and decompression MSE for each DR method. 11 different compression rates have been evaluated, in particular $L = 10$ (compressing about 95.45–95%

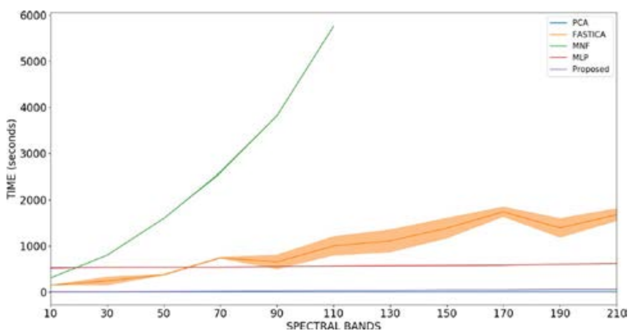


Fig. 12 Execution times of the compression experiment with the large Indian Pines scene

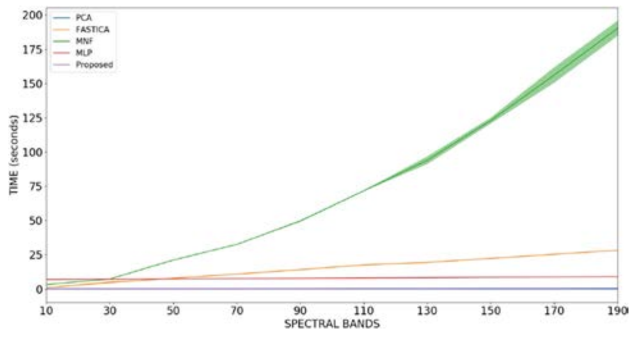


Fig. 13 Execution times of the compression experiment with the small Indian Pines scene

of the spectral bands), 30 (86.36–86%), 50 (77.27–77%), 70 (68.18–68%), 90 (59.09–59%), 110 (50–50%), 130 (40.91–41%), 150 (31.82–32%), 170 (22.73–23%), 190 (13.64–14%) and 210 (4.55–5%). In this case, NMF has been executed until $L = 110$ because from compressing the50% of spectral bands the execution times are unmanageable. Also the single hidden layer ELM reaches good performances in most cases, quite similar to NMF and MLP-AE, even better than MLP-AE in some cases (average MSE difference between NMF and ELM-AE is 3.37e−08 and between MLP-AE and ELM-AE is 5.29e−08). Again, the PCA and ICA provide the worst decompression results, being PCA the fastest compression method and the NMF the slowest one.

Figures 12 and 13 show graphically the execution times of each compression method in big and small Indian Pines with different values of L for large Indian Pines scene. As we can see in Fig. 12, the execution time of MLP-AE is very stable, between 500 and 600 s. On the other hand, PCA has an execution time that is close to zero, being our method more close to PCA than to parallel FastICA (in the firs stages) and MLP-AE, while NMF grows exponentially. We can compare this results with those presented in Fig. 13, where the same behaviour can be observed in small Indian Pines. These experiments demonstrate that the single layer ELM can be used as a fast and accurate compression method, which is very useful for the transmission of compressed hyperspectral data from sensors and computers onboard space or airborne platforms to the ground station on Earth, without losing data quality during the decompression stage even with large amounts of hyperspectral data.

4.3.2 Testing classification performance

At this point, we evaluate the classification results obtained for the hyperspectral images compressed with the proposed method. In this case, with the aim of taking compressed pixels in classification stage, for both training and testing the network, and following the procedure of the Sect. 4.3.1, hyperspectral scenes are entire compressed by the proposed method in the first place, using the 85% of the data for training. Our rationale for this experiment is that ANNs in general (and ELMs in particular) suffer from the Hughes effect when applied to hyperspectral image classification [6], mainly due to the unbalance between the high dimensionality of the hyperspectral data and the limited number of training samples available in advance. To avoid this problem, our proposed method can be used to mitigate the Hughes effect and perform a good classification. The selected topology has been described in Sect. 3.2 (see Fig. 4): four layers with sizes $d - L_1 - L_2 - m$, being d the number of spectral bands, L_1 the compression ratio (set to 40), L_2 the sparse ratio (that has been traced in ranges 1000, 2000, $\ldots$, 8000) and m the number of labels. Additionally, the regularization parameter C (see Eq. 15) has been traced from 1e−08 to 1e+08. On the other hand, in order to address the Hughes effect, different and reduced training sets have been selected, in particular 1, 3, 5, and 10% of samples per class have been randomly selected from the available labeled data for each image.

To test the proposed method, we perform a comparison between our fast approach and the ELM network described in [107]. The ELM proposed in [107] has three layers: one input layer with d neurons, being d the number of spectral bands; one hidden layer with L neurons, being L a number traced in the ranges 400, 600, 800, $\ldots$ and 2000, with parameter of regularization $C = 10^{-3}, 10^{-2}, \ldots, 10^{4}$. Also, an owner version of multinomial logistic regression (MLR), random forest (RF) and spectral convolutional neural network (1D-CNN) classifiers have been implemented and executed for this paper to compare their performance with that obtained by the proposed DR + classification method. We must remark that ELM in [107], the MLR, the RF and 1D-CNN are using non-compressed data, that means, they are taking information of the entire spectral bands, while the proposed method first compress and then classifies the hyperspectral data. This fact repercute into execution times, being the proposed method the fastest one.

Table 6 shows the obtained results using small Indian Pines and Pavia University hyperspectral scenes. As we can observe, with small Indian Pines dataset, the proposed method reaches the best accuracy results, followed by the implemented 1D-CNN classifier and the ELM proposed in [107], while the MLR and the RF presents their results so far away from the proposed method. Focusing on the proposed method and the ELM of [107], we can observe that the proposed method is 1.10 (with 5% of training) and 1.07 (with 10% of training) times better than the ELM in [107]. Even without considering the data in [107], we can observe that the proposed method reaches very good classification results with only 1% and 3% of training. On the other part, with Pavia University the proposed approach is the second

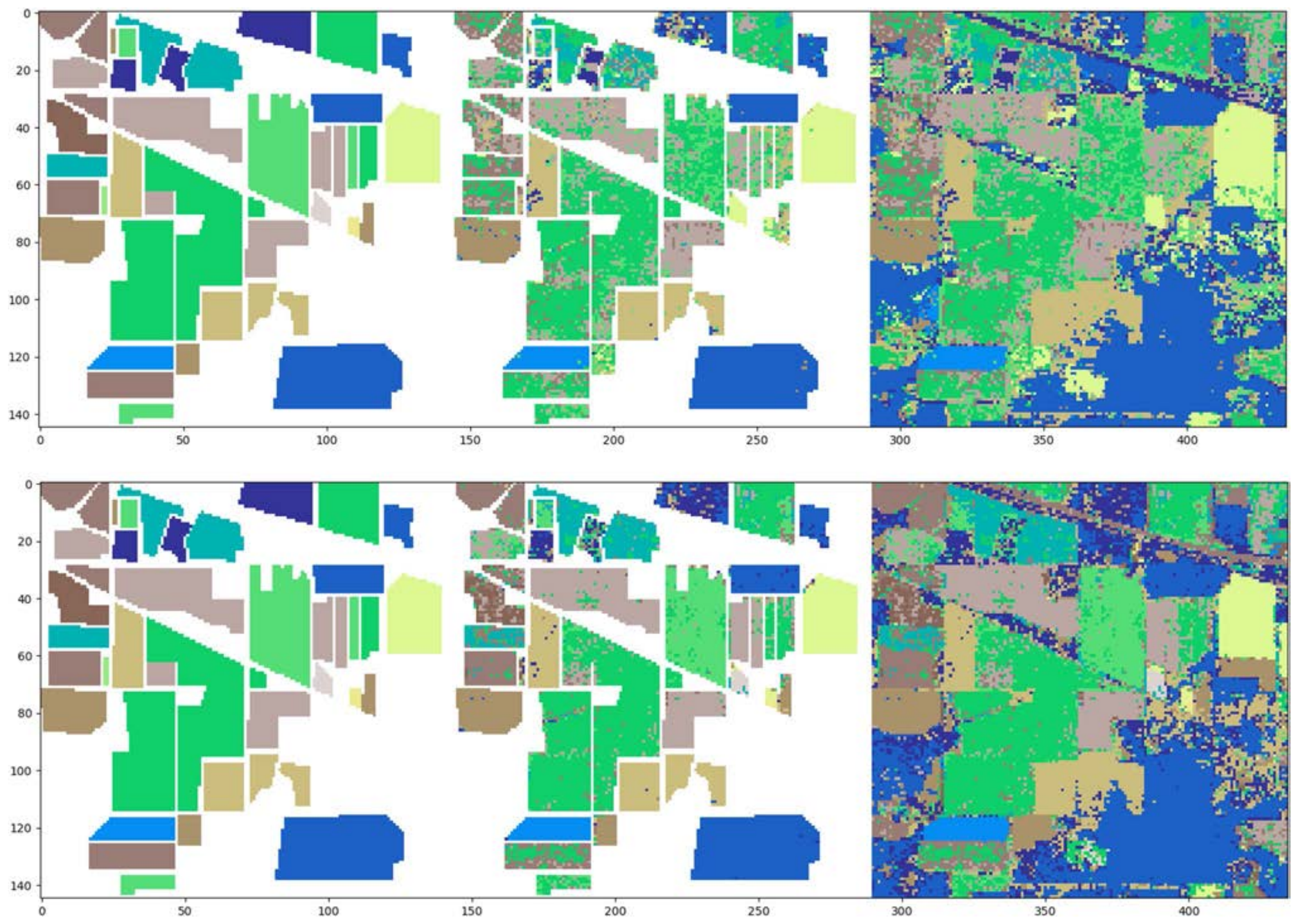


Fig. 14 Classification maps for the small Indian Pines scene with the original ground truth (left), the obtained classification map without background (center), and the obtained classification map with background (right) by the proposed method, with 1% of training (top), reaching 63.52% accuracy in 0.25 s, and with 10% of training (bottom), reaching 83.23% accuracy in 1.34 s

Table 6 Overall classification accuracy comparison (using the small Indian pines and Pavia University hyperspectral scenes) between proposed method, the MLR, RF, 1D-CNN and the ELM in [107]

Hyperspectral scene	Training (%)	MLR	RF	1D-CNN	ELM [107]	Proposed
Small Indian Pines	1	57.24 (1.41)	56.78 (1.24)	**69.74** (1.82)	–	64.91 (1.17)
	3	65.16 (1.01)	66.08 (1.49)	75.45 (1.46)	–	**75.91** (0.67)
	5	69.74 (0.61)	69.97 (0.92)	75.47 (3.03)	72,23 (1.02)	**79.77** (0.67)
	10	74.17 (0.70)	75.49 (0.85)	83.71 (0.11)	78.88 (0.88)	**84.76** (0.53)
Pavia University	1	82.22 (0.87)	82.18 (0.62)	**90.37** (0.52)	80.28 (1.00)	82.83 (0.05)
	3	85.56 (0.45)	86.31 (0.31)	**92.75** (0.26)	–	86.43 (0.13)
	5	86.97 (0.23)	88.05 (0.34)	**93.83** (0.43)	85.35 (0.37)	87.86 (0.03)
	10	88.55 (0.16)	90.57 (0.30)	**94.94** (0.27)	86.72 (0.20)	89.42 (0.04)

The proposed method has been executed ten times and the overall accuracy results are reported, with the standard deviation in the parentheses

Best values obtained are given in bold

method with best overall results, following closely the 1D-CNN classifier. Moreover, the proposed method is able to achieve better results than the ELM proposed in [107], reaching an overall accuracy which is 1.03 times better than the ELM in [107], with 1%, 5% and 10% of training. With Pavia University, MLR and RF present very similar results to proposed method, however they are using 103 spectral bands, while the proposed method uses only $L_1 = 40$ spectral bands. Finally, in both cases, with small Indian Pines and

Table 7 Classification results (accuracy) and execution times (in seconds) for the small and large Indian Pines hyperspectral datasets of PCA+MLP and the fast DR method, and Speedup reached by the proposed method in comparison with the PCA+MLP

		Classification accuracy		Execution times		Speedup
Hyperspectral scene	Training	PCA+MLP	**Proposed**	PCA+MLP	**Proposed**	
Small Indian Pines	1	58.56% (3.27)	**63.52%** (0.51)	1.01 (0.01)	**0.25** (0.01)	4.04
	3	72.01% (1.45)	**74.52%** (0.38)	3.87 (0.15)	**0.38** (0.01)	10.18
	5	**79.62%** (1.22)	79.11% (0.42)	9.53 (0.14)	**1.73** (1.16)	5.51
	10	**83.75%** (0.32)	83.23% (0.22)	14.31 (0.22)	**1.34** (1.02)	10.68
Big Indian Pines	1	**46.82%** (0.64)	46.64% (0.18)	14.99 (0.07)	**1.67** (0.20)	8.98
	2	**51.13%** (0.24)	50.13% (0.37)	29.75 (0.18)	**1.45** (0.03)	20.52
	3	52.72% (1.97)	**52.77%** (0.45)	45.12 (0.24)	**4.72** (0.05)	9.56
	5	**56.82%** (0.66)	55.43% (0.77)	75.41 (0.44)	**13.45** (0.13)	5.61
	10	**62.69%** (0.65)	59.00% (2.31)	150.46 (0.53)	**23.70** (0.32)	6.35

Both methods have been executed 10 times and the average results are reported, with the standard deviation in the parentheses

Best values obtained are given in bold

Pavia University, the proposed method is also more robust, with lower standard deviation.

Second experiment performs a comparison between the proposed fast DR and classifier and the fastest DR method, the multicore PCA compressor, whose compressed hyperspectral representation is classified by a neural classifier, implemented by the GPU MLP classifier. Again, the PCA compresses the hyperspectral image to L_1 bands, and the MLP-classifier is a single hidden layer network $L_1 - L_2 - m$, where the hidden layer expands the input data in L_2 nodes, being $L_2 > L_1$ and m the number of classes. For all experiments, L_1 has been set to 40. Table 7 shows the classification results obtained using the small and large Indian Pines hyperspectral scenes and the execution times, respectively. We can see that the proposed method can achieve very similar accuracy results to those obtained by the PCA+MLP network, reaching good results with few samples per class (e.g., 63.52% with only 1% of training for the small Indian Pines and 52.77% with 3% of training for the large Indian Pines scene) and improving the accuracy as the number of samples per class increases. However, in Table 7 we can see that the proposed method based on ELMs achieves the best execution times, being the fasted method in comparison with the PCA+MLP. From this table we can see that the proposed method not only achieves results very similar to the PCA+MLP, but it is also much faster than it, being a viable option for real-time applications. In this regard, it is worth noting that all the results reported for our method are strictly in real-time, which means that they are obtained in a time that is smaller than the data acquisition time for the considered images. Nowadays, hyperspectral data acquisition rates have been increased due to technological advances [108, 109], allowing new instruments to provide a continuous flow of data in the order of several terabytes per hour [110]. For instance, NASA is continuously gathering hyperspectral data through instruments such as the airborne Jet Propulsion Laboratory's sensor AVIRIS which comprises several GBs per flight [111], with a collection rate of 2.5 MB/s (nearly 9 GB/hour[1]). In this particular experiment, Small Indian Pines stores 8.49 MB of raw data in memory, and Big Indian Pines stores 2894 MB (both in 16 bit format). To meet real-time processing performance, the compression and classification times must be smaller than the acquisition time of the images, i.e., they must be smaller than $8.49/2.5 = 3.40$ s and $2894/2.5 = 1157.6$ s. We can conclude that proposed fast DR method is able to process and classify these amounts of data in real-time, with a compression and classification time much lower than the acquisition times of these remote sensing images.

For illustrative purposes, we represent the aforementioned results graphically in Figs. 14 and 15, where we can observe that the classification maps obtained by the proposed method with 1% of training (top), reaching 63.52% accuracy with the small Indian Pines scene and 46.64% with the large Indian Pines scene, and with 10% of training (bottom), reaching 83.23% accuracy with the small Indian Pines scene and 59.00% with the large Indian Pines scene. These classification maps are very similar to any map obtained by a MLP network. However, the execution times are much smaller (and strictly in real-time) when the proposed method is used, which is very important for practical exploitation of the considered hyperspectral images in a number of applications with time-critical constraints.

4.4 Conclusions and future research lines

In this paper, we have developed a new ELM-based dimensionality reduction and classification technique that can offer different functionalities for remotely sensed hyperspectral

[1] https://aviris.jpl.nasa.gov/aviris/instrument.html

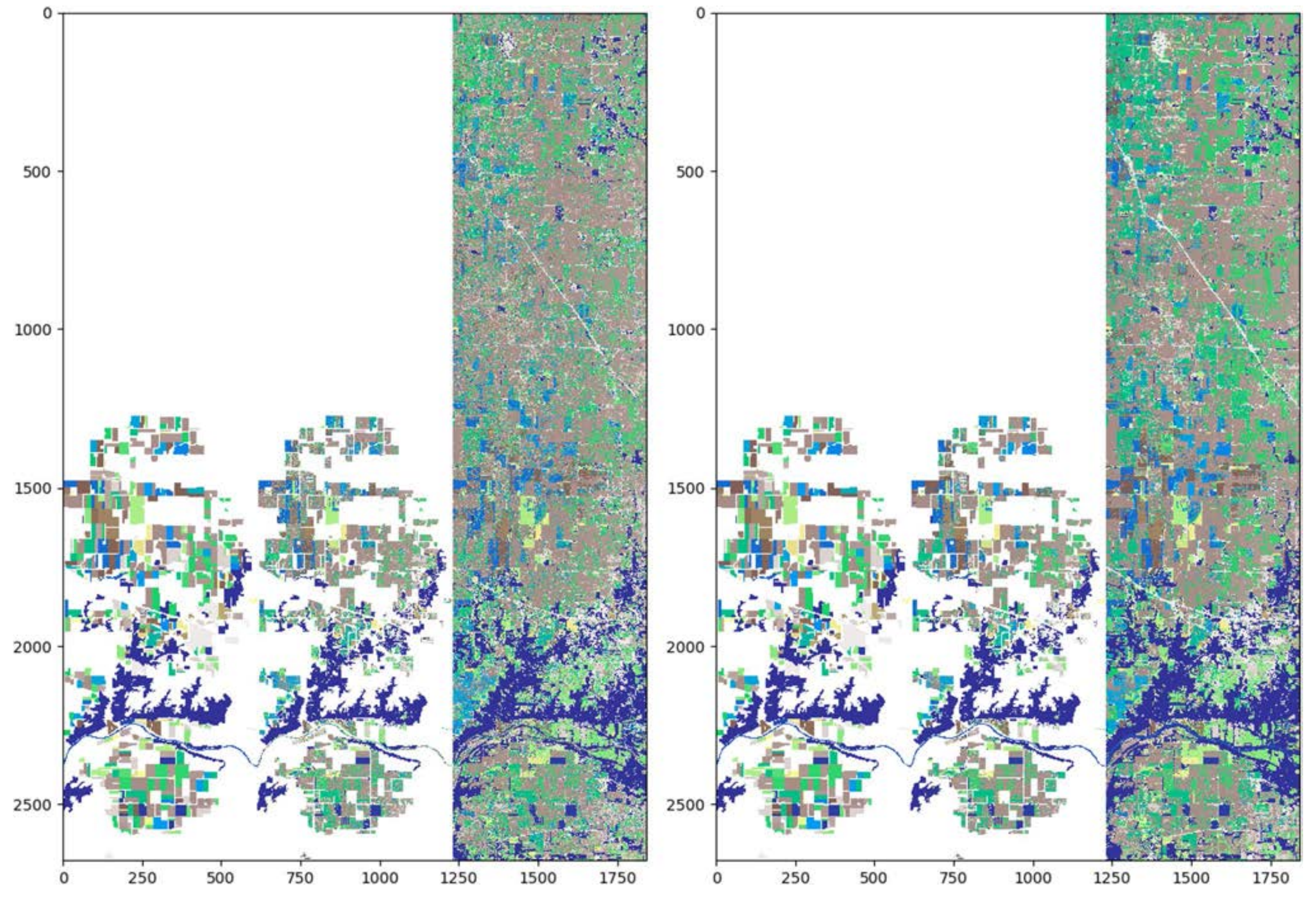


Fig. 15 Classification maps for the large Indian Pines scene with the original ground truth (left), the obtained classification map without background (center) and the obtained classification map with background (right) by the proposed method, with 1% of training (left), reaching 46.64% accuracy in 1.67 s, and with 10% of training (right), reaching 59.00% accuracy in 23.70 s

data exploitation. First and foremost, the proposed approach can compress the huge data volume of hyperspectral remote sensing data resulting from its large spectral dimensionality. In addition, the proposed method can also perform advanced classification of the compressed hyperspectral scene, providing a framework that allows fast and accurate exploitation of hyperspectral scenes since the problems motivated by their large dimensionality (e.g., the Hughes effect) can be circumvented. Our experimental results indicate that, although the compression stage of the proposed method is slower than the widely used PCA, the method can produce better compression/decompression error reconstruction, being much faster than other fast approaches such as the NMF, ICA and MLP, reaching quite similar or even lower errors in comparison with NMF and MLP (despite NMF and MLP employ a more significant computational effort). As a result, the non-iterative nature of our newly proposed single layer ELM fast approach can achieve an acceptable error without data loss in less time, offering a powerful tool for high-dimensional data transmission between the onboard platform and the ground station on Earth. In addition, we have also shown that the proposed method can reach very good classification performance when compared with other methods such as the classical ELM in [107], the MLR, RF and 1D-CNN, even when using very small amounts of training data. The proposed method can also reach similar classification results as compared to other widely used approaches, such as the PCA+MLP of one hidden layer, employing significantly less time than this method, even when analyzing very large hyperspectral scenes. In fact, the proposed approach offers important advantages for practical exploitation since the compression and classification results obtained in our experiments are strictly in real-time. Future work will focus on the development of hardware implementations in specialized platforms that can reduce even more the processing times for large-scale processing of large hyperspectral data repositories. Also, new ELM approaches will be studied for hyperspectral compression, such as the sparse ELM (SELM).

Acknowledgements This work has been supported by Ministerio de Educación (Resolución de 26 de diciembre de 2014 y de 19 de noviembre de 2015, de la Secretaría de Estado de Educación, Formación Profesional y Universidades, por la que se convocan ayudas para la formación de profesorado universitario, de los subprogramas de Formación y de Movilidad incluidos en el Programa Estatal de Promoción del Talento y su Empleabilidad, en el marco del Plan Estatal de Investigación Científica y Técnica y de Innovación 2013-2016. This work has also been supported by Junta de Extremadura (decreto 297/2014, ayudas para la realización de actividades de investigación y desarrollo tecnológico, de divulgación y de transferencia de conocimiento por los Grupos de Investigación de Extremadura, Ref. GR15005).

References

1. Xia, J., Bombrun, L., Adali, T., Berthoumieu, Y., Germain, C.: Spectral-Spatial Classification of Hyperspectral Images Using ICA and Edge-Preserving Filter via an Ensemble Strategy. IEEE Trans. Geosci. Remote Sens. **54**(8), 4971–4982 (2016). https://doi.org/10.1109/TGRS.2016.2553842
2. Chang, C.I.: Hyperspectral Imaging: Techniques for Spectral Detection and Classification. Springer, New York (2003). https://doi.org/10.1007/978-1-4419-9170-6
3. Goetz, A.F.H., Vane, G., Solomon, J.E., Rock, B.N.: Imaging Spectrometry for Earth Remote Sensing. Science **228**(4704), 1147–1153 (1985). https://doi.org/10.1126/science.228.4704.1147
4. Chutia, D., Bhattacharyya, D.K., Sarma, K.K., Kalita, R., Sudhakar, S.: Hyperspectral remote sensing classifications: a perspective survey. Trans. GIS **20**(4), 463–490 (2016). https://doi.org/10.1111/tgis.12164
5. Fukunaga, K.: Introduction to Statistical Pattern Recognition. Academic Press, Cambridge (1990)
6. Khodadadzadeh, M., Li, J., Plaza, A., Ghassemian, H., Bioucas-Dias, J.M., Li, X.: Spectral spatial classification of hyperspectral data using local and global probabilities for mixed pixel characterization. IEEE Trans. Geosci. Remote Sens. **52**(10), 6298–6314 (2014). https://doi.org/10.1109/TGRS.2013.2296031
7. Hughes, G.: On the mean accuracy of statistical pattern recognizers. IEEE Trans. Inf. Theory **14**(1), 55–63 (1968). https://doi.org/10.1109/TIT.1968.1054102
8. Wold, S., Esbensen, K., Geladi, P.: Principal component analysis. Chemometr. Intell. Lab. Syst. **2**(1), 37–52 (1987). https://doi.org/10.1016/0169-7439(87)80084-9
9. Jolliffe, I.: Principal Component Analysis. Springer, Springer Series in Statistics (2002)
10. Fernandez, D., Gonzalez, C., Mozos, D., Lopez, S.: Fpga implementation of the principal component analysis algorithm for dimensionality reduction of hyperspectral images. J. Real-Time Image Process (2016). https://doi.org/10.1007/s11554-016-0650-7
11. Fauvel, M., Chanussot, J., Benediktsson, J.A.: Kernel principal component analysis for the classification of hyperspectral remote sensing data over urban areas. EURASIP J. Adv. Signal Process. **2009**(1), 783,194 (2009). https://doi.org/10.1155/2009/783194
12. Li, Y., Wu, Z., Wei, J., Plaza, A., Li, J., Wei, Z.: Fast principal component analysis for hyperspectral imaging based on cloud computing. In: 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Milan, pp. 513–516 (2015). https://doi.org/10.1109/IGARSS.2015.7325813
13. Lin, B., Tao, G., Kai, D.: Using non-negative matrix factorization with projected gradient for hyperspectral images feature extraction. In: 2013 IEEE 8th Conference on Industrial Electronics and Applications (ICIEA), Melbourne, VIC, pp. 516–519 (2013). https://doi.org/10.1109/ICIEA.2013.6566423
14. Gillis, N., Plemmons, R.J.: Sparse nonnegative matrix underapproximation and its application to hyperspectral image analysis. Linear Algebra Appl. **438**(10), 3991–4007 (2013). https://doi.org/10.1016/j.laa.2012.04.033. (**Special issue in honor of Abraham Berman, Moshe Goldberg, and Raphael Loewy**)
15. Villa, A., Chanussot, J., Jutten, C., Benediktsson, J.A., Moussaoui, S.: On the use of ICA for hyperspectral image analysis. In: 2009 IEEE International Geoscience and Remote Sensing Symposium, Cape Town, pp. IV-97–IV-100 (2009). https://doi.org/10.1109/IGARSS.2009.5417363
16. Villa, A., Benediktsson, J.A., Chanussot, J., Jutten, C.: Hyperspectral image classification with Independent component discriminant analysis. IEEE Trans. Geosci. Remote Sens. **49**(12), 4865–4876 (2011). https://doi.org/10.1109/TGRS.2011.2153861
17. Wang, J., Chang, C.I.: Independent component analysis-based dimensionality reduction with applications in hyperspectral image analysis. IEEE Trans. Geosci. Remote Sens. **44**(6), 1586–1600 (2006). https://doi.org/10.1109/TGRS.2005.863297
18. Green, A.A., Berman, M., Switzer, P., Craig, M.D.: A transformation for ordering multispectral data in terms of image quality with implications for noise removal. IEEE Trans. Geosci. Remote Sens. **26**(1), 65–74 (1988). https://doi.org/10.1109/36.3001
19. Chang, C.I., Du, Q.: Interference and noise-adjusted principal components analysis. IEEE Trans. Geosci. Remote Sens. **37**(5), 2387–2396 (1999). https://doi.org/10.1109/36.789637
20. Lee, J.B., Woodyatt, A.S., Berman, M.: Enhancement of high spectral resolution remote-sensing data by a noise-adjusted principal components transform. IEEE Trans. Geosci. Remote Sens. **28**(3), 295–304 (1990). https://doi.org/10.1109/36.54356
21. Iyer, R.P., Raveendran, A., Bhuvana, S.K.T., Kavitha, R.: Hyperspectral image analysis techniques on remote sensing. In: 2017 Third International Conference on Sensing, Signal Processing and Security (ICSSS), Chennai, pp. 392–396 (2017). https://doi.org/10.1109/SSPS.2017.8071626
22. Kuybeda, O., Malah, D., Barzohar, M.: Rank estimation and redundancy reduction of high-dimensional noisy signals with preservation of rare vectors. IEEE Trans. Signal Process. **55**(12), 5579–5592 (2007). https://doi.org/10.1109/TSP.2007.901645
23. Acito, N., Diani, M., Corsini, G.: A new algorithm for robust estimation of the signal subspace in hyperspectral images in the presence of rare signal components. IEEE Trans. Geosci. Remote Sens. **47**(11), 3844–3856 (2009). https://doi.org/10.1109/TGRS.2009.2021764
24. Acito, N., Diani, M., Corsini, G.: Hyperspectral signal subspace identification in the presence of rare signal components. IEEE Trans. Geosci. Remote Sens. **48**(4), 1940–1954 (2010). https://doi.org/10.1109/TGRS.2009.2035445
25. Acito, N., Diani, M., Corsini, G.: Hyperspectral signal subspace identification in the presence of rare vectors and signal-dependent noise. IEEE Trans. Geosci. Remote Sens. **51**(1), 283–299 (2013). https://doi.org/10.1109/TGRS.2012.2201488
26. Atkinson, P.M., Tatnall, A.R.L.: Introduction Neural networks in remote sensing. Int. J. Remote Sens. **18**(4), 699–709 (1997). https://doi.org/10.1080/014311697218700
27. Benediktsson, J.A., Swain, P.H., Ersoy, O.K.: Conjugate gradient neural networks in classification of very high dimensional remote sensing data. Int. J. Remote Sens. **14**(15), 2883–2903 (1993)
28. Paoletti, M.E., Haut, J.M., Plaza, J., Plaza, A.: A new deep convolutional neural network for fast hyperspectral image classification. ISPRS J. Photogramm. Remote Sens. (2017). https://doi.org/10.1016/j.isprsjprs.2017.11.021
29. Bishop, C.: Pattern Recognition and Machine Learning. Springer-Verlag, New York (2006)

30. Hinton, G.E., Zemel, R.S.: Autoencoders, minimum description length and helmholtz free energy. In: Proceedings of the 6th International Conference on Neural Information Processing Systems, NIPS'93, pp. 3–10. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1993)
31. Bishop, C.: Neural Networks for Pattern Recognition. Advanced Texts in Econometrics. Clarendon Press, New York (1995)
32. Hinton, G., Salakhutdinov, R.: Reducing the dimensionality of data with neural networks. Science **313**, 504–507 (2006). https://doi.org/10.1126/science.1127647
33. Chen, Y., Lin, Z., Zhao, X., Wang, G., Gu, Y.: Deep learning-based classification of hyperspectral data. IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens. **7**(6), 2094–2107 (2014). https://doi.org/10.1109/JSTARS.2014.2329330
34. Karhunen, J., Raiko, T., Cho, K.H.: Chapter 7—Unsupervised deep learning: a short review. In: Advances in Independent Component Analysis and Learning Machines, pp. 125–142. Academic Press (2015). ISBN 9780128028063. https://doi.org/10.1016/B978-0-12-802806-3.00007-5
35. Zhang, P., Gong, M., Su, L., Liu, J., Li, Z.: Change detection based on deep feature representation and mapping transformation for multi-spatial-resolution remote sensing images. ISPRS J. Photogramm. Remote Sens. **116**, 24–41 (2016). https://doi.org/10.1016/j.isprsjprs.2016.02.013
36. Licciardi, G.A., Chanussot, J., Piscini, A.: Spectral compression of hyperspectral images by means of nonlinear principal component analysis decorrelation. In: 2014 IEEE International Conference on Image Processing (ICIP), pp. 5092–5096 (2014). https://doi.org/10.1109/ICIP.2014.7026031
37. Cavalli, R.M., Licciardi, G.A., Chanussot, J.: Detection of anomalies produced by buried archaeological structures using nonlinear principal component analysis applied to airborne hyperspectral image. IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens. **6**(2), 659–669 (2013). https://doi.org/10.1109/JSTARS.2012.2227301
38. Penalver, M., Del Frate, F., Paoletti, M.E., Haut, J.M., Plaza, J., Plaza, A.: Onboard payload-data dimensionality reduction. In: 2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Fort Worth, TX, pp. 783–786 (2017). https://doi.org/10.1109/IGARSS.2017.8127069
39. Benediktsson, J.A., Swain, P.H.: Statistical Methods and Neural Network Approaches for Classification of Data from Multiple Sources. Ph.D. thesis, PhD thesis, Purdue Univ., School of Elect. Eng. West Lafayette, IN (1990)
40. Haut, J.M., Paoletti, M., Plaza, J., Plaza, A.: Evaluación del rendimiento de una implementación Cloud para un clasificador neuronal aplicado a imágenes hiperespectrales. Actas Jornadas Sarteco pp. 127–134 (2016)
41. Richards, J.A.: Analysis of remotely sensed data: the formative decades and the future. IEEE Trans. Geosci. Remote Sens. **43**(3), 422–432 (2005). https://doi.org/10.1109/TGRS.2004.837326
42. Carlsohn, M.: Special issue on spectral imaging: Real-time processing of hyperspectral data. J. Real-Time Image Proc. **1**(2), 99–100 (2006). https://doi.org/10.1007/s11554-006-0020-y
43. Plaza, A.J.: Preface to the Special issue on architectures and techniques for real-time processing of remotely sensed images. J. Real-Time Image Proc. **4**(3), 191–193 (2009). https://doi.org/10.1007/s11554-009-0126-0
44. du, Q., Nekovei, R.: Fast real-time onboard processing of hyperspectral imagery for detection and classification. J. Real-Time Image Process. **4**(3), 273–286 (2009). https://doi.org/10.1007/s11554-008-0106-9
45. Huang, G.B., Zhu, Q.Y., Siew, C.K.: Extreme learning machine: theory and applications. Neurocomputing **70**(13), 489–501 (2006). https://doi.org/10.1016/j.neucom.2005.12.126. Neural NetworksSelected Papers from the 7th Brazilian Symposium on Neural Networks (SBRN '04)7th Brazilian Symposium on Neural Networks
46. Samat, A., Du, P., Liu, S., Li, J., Cheng, L.: E^2LMs: ensemble extreme learning machines for hyperspectral image classification. IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens. **7**(4), 1060–1069 (2014). https://doi.org/10.1109/JSTARS.2014.2301775
47. Pal, M.: Extreme-learning-machine-based land cover classification. Int. J. Remote Sens. **30**(14), 3835–3841 (2009). https://doi.org/10.1080/01431160902788636
48. Pal, M., Maxwell, A.E., Warner, T.A.: Kernel-based extreme learning machine for remote-sensing image classification. Remote Sens. Lett. **4**(9), 853–862 (2013). https://doi.org/10.1080/2150704X.2013.805279
49. Chen, C., Li, W., Su, H., Liu, K.: Spectral-spatial classification of hyperspectral image based on kernel extreme learning machine. Remote Sens. **6**(6), 5795–5814 (2014). https://doi.org/10.3390/rs6065795
50. Zhou, Y., Peng, J., Chen, C.L.P.: Extreme learning machine with composite kernels for hyperspectral image classification. IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens. **8**(6), 2351–2360 (2015). https://doi.org/10.1109/JSTARS.2014.2359965
51. Lv, Q., Niu, X., Dou, Y., Wang, Y., Xu, J., Zhou, J.: Hyperspectral image classification via kernel extreme learning machine using local receptive fields. In: 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, pp. 256–260 (2016). https://doi.org/10.1109/ICIP.2016.7532358
52. Shen, Y., Xu, J., Li, H., Xiao, L.: ELM-based spectral-spatial classification of hyperspectral images using bilateral filtering information on spectral band-subsets. In: 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Beijing, pp. 497–500 (2016). https://doi.org/10.1109/IGARSS.2016.7729123
53. Shen, Y., Chen, J., Xiao, L.: Supervised classification of hyperspectral images using local-receptive-fields-based kernel extreme learning machine. In: 2017 IEEE International Conference on Image Processing (ICIP), Beijing, pp. 3120–3124 (2017). https://doi.org/10.1109/ICIP.2017.8296857
54. Bazi, Y., Alajlan, N., Melgani, F., AlHichri, H., Malek, S., Yager, R.R.: Differential evolution extreme learning machine for the classification of hyperspectral images. IEEE Geosci. Remote Sens. Lett. **11**(6), 1066–1070 (2014). https://doi.org/10.1109/LGRS.2013.2286078
55. Heras, D.B., Argüello, F., Quesada-Barriuso, P.: Exploring elm-based spatial-spectral classification of hyperspectral images. Int. J. Remote Sens. **35**(2), 401–423 (2014). https://doi.org/10.1080/01431161.2013.869633
56. Moreno, R., Corona, F.: Lendasse, A., Graña, M.G., Galvão, L.S.G.: Extreme learning machines for soybean classification in remote sensing hyperspectral images. Neurocomputing 128, 207–216 (2014). https://doi.org/10.1016/j.neucom.2013.03.057
57. Yan, D., Chu, Y., Li, L., Liu, D.: Hyperspectral remote sensing image classification with information discriminative extreme learning machine. Multimed. Tools Appl. **77**(5), 5803–5818 (2018). https://doi.org/10.1007/s11042-017-4494-3
58. Xu, J., Li, H., Liu, P., Xiao, L.: A novel hyperspectral image clustering method with context-aware unsupervised discriminative extreme learning machine. IEEE Access **PP**(99), 1–1 (2018). https://doi.org/10.1109/ACCESS.2018.2813988
59. Chen, H., Peng, J., Zhou, Y., Li, L., Pan, Z.: Extreme learning machine for ranking: generalization analysis and applications. Neural Netw. **53**, 119–126 (2014). https://doi.org/10.1016/j.neunet.2014.01.015
60. Li, J., Kingsdorf, B., Du, Q.: Band selection for hyperspectral image classification using extreme learning machine. In: Proc. SPIE 10198, Algorithms and Technologies for Multispectral,

Hyperspectral, and Ultraspectral Imagery XXIII, 101980R, 5 May 2017. https://doi.org/10.1117/12.2263039

61. Alhichri, H., Bazi, Y., Alajlan, N., Ammour, N.: A hierarchical learning paradigm for semi-supervised classification of remote sensing images. In: 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Milan, pp. 4388–4391 (2015). https://doi.org/10.1109/IGARSS.2015.7326799
62. Su, H., Cai, Y.: Firefly algorithm optimized extreme learning machine for hyperspectral image classification. In: 2015 23rd International Conference on Geoinformatics, Wuhan, pp. 1–4 (2015). https://doi.org/10.1109/GEOINFORMATICS.2015.7378645
63. Basterretxea, K., Martinez-Corral, U., Finker, R., del Campo, I.: Elm-based hyperspectral imagery processor for onboard real-time classification. In: 2016 Conference on Design and Architectures for Signal and Image Processing (DASIP), pp. 43–50 (2016). https://doi.org/10.1109/DASIP.2016.7853795
64. Cambria, E., Huang, G.B., Kasun, L.L.C., Zhou, H., Vong, C.M., Lin, J., Yin, J., Cai, Z., Liu, Q., Li, K., Leung, V.C.M., Feng, L., Ong, Y.S., Lim, M.H., Akusok, A., Lendasse, A., Corona, F., Nian, R., Miche, Y., Gastaldo, P., Zunino, R., Decherchi, S., Yang, X., Mao, K., Oh, B.S., Jeon, J., Toh, K.A., Teoh, A.B.J., Kim, J., Yu, H., Chen, Y., Liu, J.: Extreme learning machines [trends controversies]. IEEE Intell. Syst. **28**(6), 30–59 (2013). https://doi.org/10.1109/MIS.2013.140
65. Zhou, H., Huang, G.B., Lin, Z., Wang, H., Soh, Y.C.: Stacked extreme learning machines. IEEE Trans. Cybern. **45**(9), 2013–2025 (2015). https://doi.org/10.1109/TCYB.2014.2363492
66. Kasun, L.L.C., Yang, Y., Huang, G.B., Zhang, Z.: Dimension reduction with extreme learning machine. IEEE Trans. Image Process. **25**(8), 3906–3918 (2016). https://doi.org/10.1109/TIP.2016.2570569
67. Jutten, C., Herault, J.: Blind separation of sources, part i: an adaptive algorithm based on neuromimetic architecture. Sig. Process. **24**(1), 1–10 (1991). https://doi.org/10.1016/0165-1684(91)90079-X
68. Comon, P., Jutten, C.: Handbook of Blind Source Separation: Independent Component Analysis and Applications. Academic press, Cambridge (2010)
69. Santamaria, I.: Handbook of blind source separation: independent component analysis and applications (common, p. and jutten,; 2010 [book review]. IEEE Signal Process. Mag. **30**(2), 133–134 (2013). https://doi.org/10.1109/MSP.2012.2230552
70. Falco, N., Bruzzone, L., Benediktsson, J.A.: A comparative study of different ICA algorithms for hyperspectral image analysis. In: 2013 5th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS), Gainesville, FL, pp. 1–4 (2013). https://doi.org/10.1109/WHISPERS.2013.8080596
71. Falco, N., Benediktsson, J.A., Bruzzone, L.: A study on the effectiveness of different independent component analysis algorithms for hyperspectral image classification. IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens. **7**(6), 2183–2199 (2014). https://doi.org/10.1109/JSTARS.2014.2329792
72. Villa, A., Benediktsson, J.A., Chanussot, J., Jutten, C.: Independent Component Discriminant Analysis for hyperspectral image classification. In: 2010 2nd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing, Reykjavik, pp. 1–4 (2010). https://doi.org/10.1109/WHISPERS.2010.5594853
73. Mura, M.D., Villa, A., Benediktsson, J.A., Chanussot, J., Bruzzone, L.: Classification of hyperspectral images by using extended morphological attribute profiles and independent component analysis. IEEE Geosci. Remote Sens. Lett. **8**(3), 542–546 (2011). https://doi.org/10.1109/LGRS.2010.2091253
74. Nascimento, J.M.P., Dias, J.M.B.: Does independent component analysis play a role in unmixing hyperspectral data? IEEE Trans. Geosci. Remote Sens. **43**(1), 175–187 (2005). https://doi.org/10.1109/TGRS.2004.839806
75. Hyvrinen, A., Oja, E.: Independent component analysis: algorithms and applications. Neural Netw. **13**(4), 411–430 (2000). https://doi.org/10.1016/S0893-6080(00)00026-5
76. Green, A.A., Berman, M., Switzer, P., Craig, M.D.: A transformation for ordering multispectral data in terms of image quality with implications for noise removal. IEEE Trans. Geosci. Remote Sens. **26**(1), 65–74 (1988). https://doi.org/10.1109/36.3001
77. Boutsidis, C., Gallopoulos, E.: Svd based initialization: a head start for nonnegative matrix factorization. Pattern Recogn. **41**(4), 1350–1362 (2008). https://doi.org/10.1016/j.patcog.2007.09.010
78. Huang, G.-B., Siew, C.-K.: Extreme learning machine: RBF network case. In: ICARCV 2004 8th Control, Automation, Robotics and Vision Conference, vol. 2, pp. 1029–1036 (2004). https://doi.org/10.1109/ICARCV.2004.1468985
79. Huang, G.B., Chen, L., Siew, C.K.: Universal approximation using incremental constructive feedforward networks with random hidden nodes. Trans. Neural Netw. **17**(4), 879–892 (2006). https://doi.org/10.1109/TNN.2006.875977
80. Ding, S., Xu, X., Nie, R.: Extreme learning machine and its applications. Neural Comput. Appl. **25**(3), 549–556 (2014). https://doi.org/10.1007/s00521-013-1522-8
81. Huang, G.B., Zhou, H., Ding, X., Zhang, R.: Extreme learning machine for regression and multiclass classification. IEEE Trans. Syst. Man Cybern. **42**(2), 513–529 (2012). https://doi.org/10.1109/TSMCB.2011.2168604
82. Tang, J., Deng, C., Huang, G.B.: Extreme learning machine for multilayer perceptron. IEEE Trans. Neural Netw. Learn. Syst. **27**(4), 809–821 (2016). https://doi.org/10.1109/TNNLS.2015.2424995
83. Huang, G., Liu, T., Yang, Y., Lin, Z., Song, S., Wu, C.: Discriminative clustering via extreme learning machine. Neural Netw. **70**(Supplement C), 1–8 (2015). https://doi.org/10.1016/j.neunet.2015.06.002
84. Huang, G., Song, S., Gupta, J.N.D., Wu, C.: Semi-supervised and unsupervised extreme learning machines. IEEE Trans. Cybern. **44**(12), 2405–2417 (2014). https://doi.org/10.1109/TCYB.2014.2307349
85. Zhu, W., Miao, J., Qing, L.: Constrained extreme learning machines: a study on classification cases (2015). arXiv:1501.06115
86. Kasun, L.L.C., Zhou, H., Huang, G.B., Vong, C.M.: Representational learning with elms for big data. IEEE Intell. Syst. **28**(6), 31–34 (2013)
87. Huang, G.B.: What are extreme learning machines? Filling the gap between frank rosenblatts dream and john von neumanns puzzle. Cogn. Comput. **7**, 263278 (2015). https://doi.org/10.1007/s12559-015-9333-0
88. Huang, G.B., Bai, Z., Kasun, L.L.C., Vong, C.M.: Local receptive fields based extreme learning machine. IEEE Comput. Intell. Mag. **10**(2), 18–29 (2015). https://doi.org/10.1109/MCI.2015.2405316
89. Huang, G.B., Chen, L.: Convex incremental extreme learning machine. Neurocomputing **70**(16), 3056–3062 (2007). https://doi.org/10.1016/j.neucom.2007.02.009. Artificial Neural Networks (IWANN 2005)
90. Huang, G.B., Chen, L.: Enhanced random search based incremental extreme learning machine. Neurocomputing **71**(16), 3460–3468 (2008). https://doi.org/10.1016/j.neucom.2007.10.008. Advances in Neural Information Processing (ICONIP 2006) / Brazilian Symposium on Neural Networks (SBRN 2006)
91. Zhang, R., Lan, Y., Huang, G.B., Xu, Z.B.: Universal approximation of extreme learning machine with adaptive growth of hidden

nodes. IEEE Trans. Neural Netw. Learn. Syst. **23**(2), 365–371 (2012). https://doi.org/10.1109/TNNLS.2011.2178124

92. Huang, G.B.: An insight into extreme learning machines: random neurons, random features and kernels. Cogn. Comput. **6**(3), 376–390 (2014). https://doi.org/10.1007/s12559-014-9255-2
93. Tamura, S., Tateishi, M.: Capabilities of a four-layered feedforward neural network: four layers versus three. IEEE Trans. Neural Netw. **8**(2), 251–255 (1997). https://doi.org/10.1109/72.557662
94. Huang, G.B., Zhu, Q.Y., Siew, C.K.: Real-time learning capability of neural networks. IEEE Trans. Neural Netw. **17**(4), 863–878 (2006). https://doi.org/10.1109/TNN.2006.875974
95. Huang, G.-B., Zhu, Q.-Y., Siew, C.-K.: Extreme learning machine: a new learning scheme of feedforward neural networks. In: 2004 IEEE International Joint Conference on Neural Networks, vol. 2, pp. 985–990. IEEE Cat. No.04CH37541 (2004). https://doi.org/10.1109/IJCNN.2004.1380068
96. Bartlett, P.L.: The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network. IEEE Trans. Inf. Theory **44**(2), 525–536 (1998). https://doi.org/10.1109/18.661502
97. Zhu, Q.Y., Qin, A., Suganthan, P., Huang, G.B.: Evolutionary extreme learning machine. Pattern Recogn. **38**(10), 1759–1763 (2005). https://doi.org/10.1016/j.patcog.2005.03.028
98. Rao, C., Mitra, S.K.: Generalized Inverse of Matrices and Its Applications. Wiley Probability and Statistics Series, New York (1971)
99. Ben-Israel, A., Greville, T.N.E.: Generalized Inverses: Theory and Applications. Springer, New York (2003). https://doi.org/10.1007/b97366
100. Campbell, S., Meyer, C.: Generalized Inverses of Linear Transformations. Society for Industrial and Applied Mathematics (2009). https://doi.org/10.1137/1.9780898719048
101. Xin, J., Wang, Z., Qu, L., Wang, G.: Elastic extreme learning machine for big data classification. Neurocomputing **149**(Part A), 464–471 (2015). https://doi.org/10.1016/j.neucom.2013.09.075. (**Advances in neural networks Advances in Extreme Learning Machines**)
102. Peng, Y., Kong, W., Yang, B.: Orthogonal extreme learning machine for image classification. Neurocomputing **266**(Supplement C), 458–464 (2017). https://doi.org/10.1016/j.neucom.2017.05.058
103. Green, R.O., Eastwood, M.L., Sarture, C.M., Chrien, T.G., Aronsson, M., Chippendale, B.J., Faust, J.A., Pavri, B.E., Chovit, C.J., Solis, M., Olah, M.R., Williams, O.: Imaging spectroscopy and the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS). Remote Sens. Environ. **65**(3), 227–248 (1998). https://doi.org/10.1016/S0034-4257(98)00064-9
104. Kunkel, B., Blechinger, F., Lutz, R., Doerffer, R., van der Piepen, H., Schroder, M.: ROSIS (Reflective Optics System Imaging Spectrometer)—a candidate instrument for polar platform missions. In: Proc. SPIE 0868, Optoelectronic Technologies for Remote Sensing from Space, 13 April 1988. https://doi.org/10.1117/12.943611
105. Xu, X., Li, J., Plaza, A.: Fusion of hyperspectral and LiDAR data using morphological component analysis. In: 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Beijing, pp. 3575–3578 (2016). https://doi.org/10.1109/IGARSS.2016.7729926
106. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization (2014). arXiv:1412.6980
107. Ghamisi, P., Plaza, J., Chen, Y., Li, J., Plaza, A.J.: Advanced spectral classifiers for hyperspectral images: a review. In: IEEE Geoscience and Remote Sensing Magazine, vol. 5, no. 1, pp. 8–32, March 2017. https://doi.org/10.1109/MGRS.2016.2616418
108. Buckner, J.L.: NASA Advanced Component Technology Program, investments in remote sensing technologies. In: Proceedings of 2003 IEEE International Geoscience and Remote Sensing Symposium (IGARSS 2003), vol. 1, pp. 494–496. IEEE Cat. No. 03CH37477 (2003). https://doi.org/10.1109/IGARSS.2003.1293820
109. Lucas, R., Rowlands, A., Niemann, O., Merton, R.: Advanced Image Processing Techniques for Remotely Sensed Hyperspectral Data. Springer Berlin Heidelberg, Berlin, Heidelberg (2004). https://doi.org/10.1007/978-3-662-05605-9-2
110. Plaza, J., Perez, R., Plaza, A., Martinez, P., Valencia, D.: Parallel morphological/neural classification of remote sensing images using fully heterogeneous and homogeneous commodity clusters. In: 2006 IEEE International Conference on Cluster Computing, Barcelona, pp. 1–10 (2006). https://doi.org/10.1109/CLUSTR.2006.311867
111. Sánchez, S., Ramalho, R., Sousa, L., Plaza, A.: Real-time implementation of remotely sensed hyperspectral image unmixing on gpus. J. Real-Time Image Proc. **10**(3), 469–483 (2015). https://doi.org/10.1007/s11554-012-0269-2