End-to-End Convolutional Autoencoder for Nonlinear Hyperspectral Unmixing

Mohamad Dhaini, Maxime Berar, Paul Honeine, Antonin van Exem

remote sensing

MDPI

Article

End-to-End Convolutional Autoencoder for Nonlinear Hyperspectral Unmixing

Mohamad Dhaini [1,2,*], Maxime Berar [1], Paul Honeine [1] and Antonin Van Exem [2]

[1] LITIS Lab, Université de Rouen Normandie, 76000 Rouen, France; maxime.berar@univ-rouen.fr (M.B.); paul.honeine@univ-rouen.fr (P.H.)
[2] Tellux Company, 76000 Rouen, France; antonin.vanexem@tellux.fr
* Correspondence: mohamad.dhaini@univ-rouen.fr

Abstract: Hyperspectral Unmixing is the process of decomposing a mixed pixel into its pure materials (endmembers) and estimating their corresponding proportions (abundances). Although linear unmixing models are more common due to their simplicity and flexibility, they suffer from many limitations in real world scenes where interactions between pure materials exist, which paved the way for nonlinear methods to emerge. However, existing methods for nonlinear unmixing require prior knowledge or an assumption about the type of nonlinearity, which can affect the results. This paper introduces a nonlinear method with a novel deep convolutional autoencoder for blind unmixing. The proposed framework consists of a deep encoder of successive small size convolutional filters along with max pooling layers, and a decoder composed of successive 2D and 1D convolutional filters. The output of the decoder is formed of a linear part and an additive non-linear one. The network is trained using the mean squared error loss function. Several experiments were conducted to evaluate the performance of the proposed method using synthetic and real airborne data. Results show a better performance in terms of abundance and endmembers estimation compared to several existing methods.

Keywords: convolutional neural network; autoencoder; hyperspectral imaging; nonlinear spectral unmixing

Citation: Dhaini, M.; Berar, M.; Honeine, P.; Van Exem, A. End-to-End Convolutional Autoencoder for Nonlinear Hyperspectral Unmixing. *Remote Sens.* **2022**, *14*, 3341. https://doi.org/10.3390/rs14143341

Academic Editor: Paul Scheunders

Received: 25 May 2022
Accepted: 7 July 2022
Published: 11 July 2022

1. Introduction

Hyperspectral imagery delivers efficient information about the physical characteristics of an object or an area from a distance without any contact. Hyperspectral sensors provide a wide range of information in the light spectrum, with hundreds of contiguous bands in a wavelength region often between 500 nm and 2500 nm. Therefore, it gives each material its unique spectral signature.

Due to the trade-off between spectral and spatial resolutions in imaging sensors, hyperspectral images (HSI) generally suffer from a low spatial resolution and consequently the presence of pixels with mixed spectral information. Such pixels in HSI come as a mixture of more than one pure material. These pure materials in an image are referred to as endmembers, and their proportions in each mixed pixel as abundances. Hyperspectral unmixing (HSU) is the process of finding these endmembers and their abundances [1], assuming their number is either known or estimated [2].

In HSU, there are two ways to model the mixing process. First, the linear mixing model (LMM) assumes that a mixed spectrum can be decomposed into a linear combination between endmembers and their fractional abundances, while neglecting higher order interactions. Some of these models start with extracting the endmembers through geometrical methods, such as N-finder algorithm (N-FINDR) [3] or vertex component analysis (VCA) [4], followed by a sparse coding step for abundance estimation as in the fully constrained least squares (FCLS) method [5] or using a constrained optimization with

the augmented Lagrangian method of multipliers (SUnSAL) [6]. Other models extract simultaneously the endmembers and their abundances, such as with non-negative matrix factorization [7] or with geometry [8]. The LMM is the most widely used due to its simplicity and interpretability. The second way is through a nonlinear mixing model (NMM) that tries to mimic the actual mechanism of the capturing sensors by introducing nonlinear interactions between the individual materials [9]. Mostly considered NMM rely on physical phenomena as in the bilinear model [10], which introduces a second order of interactions between endmembers, the post-nonlinear mixture model (PNMM) [11], which combines linear and quadratic functions of the abundances, and the linear/nonlinear fluctuation model [12], where the nonlinear component is defined in a reproducing kernel Hilbert space with manual selection of the kernel based on the application context. However, the major downside of the existing NMMs is, besides the high computational complexity, their need for prior knowledge on the inherited nonlinearity or endmembers. There have been some attempts to circumvent this drawback by combining these predefined nonlinearities through the investigation of the residual component and defining an additive term to account for nonlinearities, endmember variabilities and mismodeling effects [13].

The use of deep learning for the unmixing problem has recently received the attention of researchers after its satisfying results in hyperspectral image classification [14,15]. This is the case for instance in [16–18], where the unmixing is carried out in a supervised scenario with fractional abundances being extracted with prior knowledge about the endmembers. Other studies perform a blind unmixing by the means of deep neural networks, such as in [19,20], where pixel-based autoencoder architectures are introduced with a one layer shallow linear decoder. However, these approaches do not take into account the underlying physics in the nonlinear mixing, namely the bilinearity, post-nonlinearity or linear/nonlinear fluctuations. To solve this, Wang et al. introduced in [21] a post-nonlinear architecture in the decoder part of the autoencoder, where two fully-connected layers were added after the linear output while maintaining a similar encoder architecture. Due to the failure of this architecture to generalize over different types of nonlinearities, Zhao et al. introduced in [22] an additive fluctuation component to the linear mixture in the decoder part by means of fully-connected layers. Thus, the nonlinear output at a specific wavelength can come as a weighted sum of the interactions between endmembers at every single wavelength.

The objective of this paper is to present a novel method for nonlinear spectral unmixing, by designing an end-to-end convolutional autoencoder architecture. It addresses the weaknesses in [18] with respect to handling linear applications only and [22] with respect to using fully-connected linear layers to represent nonlinearities. Our contributions can be summarized as follows:

1. A novel architecture of fully 1D-convolutional encoder is introduced to provide a faster and better abundance estimation. The 6-layer deep encoder with filters of small kernel sizes allows to achieve better abundances estimations in highly mixed scenarios.. One of the main advantages of convolutional layers is the ability to achieve shift and deformation invariant features. This is done using three main ideas: local receptive fields, shared weights, and signal subsampling [23]. The use of shared weights reduces the number of parameters and thus helps in better generalization. These advantages are not found in fully-connected networks, which suffer from a high number of parameters, thus requiring large-scale training datasets or pushing designers to incorporate some regularizations/pruning strategies as in [22].
2. A new nonlinear decoder architecture that utilizes 2D and 1D convolutions to provide an additive fluctuation nonlinearity without any prior knowledge. The new decoder architecture is robust to nonlinearity and reduces the confusion between endmember's abundances. The difference with [22] is that we use convolutional layers. Our motivation is that the convolutions take advantage of the correlation between neighbor spectral bands to estimate the nonlinearities of the unmixing process. This motivation

is supported by the conducted experiments, demonstrating how convolutional layers outperform fully-connected layers.

The rest of the paper is organized as follows. Section 2 presents the mathematical background of both linear and nonlinear unmixing models. In Section 2.4, the proposed model is presented as well as the training parameters. Section 3 shows the different experiments done on both synthetic and real datasets. Finally, Section 4 concludes the whole paper.

2. Method

In this section, we provide the methodology and mathematical background of the unmixing problems, while focusing on their formulations using autoencoders.

For a hyperspectral image $X \in \mathbb{R}_+^{L\times N}$, with N being the number of pixels and L the number of spectral bands, let $x_i \in \mathbb{R}_+^L$ be an observed column pixel of X. Let $M \in \mathbb{R}^{L\times R}$ be the endmember mixing matrix where R is the number of endmembers. Let $A \in \mathbb{R}_+^{R\times N}$ contain all abundance vectors, with $a_i \in \mathbb{R}_+^R$ being the abundance vector associated with the observed column pixel. Each component of a_i is non-negative (denoted ANC for abundance non-negativity constraint) and all components sum to one, namely
$1_R^\top a_i = 1$ (denoted ASC for abundance sum-to-one constraint), where 1_R is the column vector of R ones. All mathematical symbols were summarized in Abbreviations.

2.1. Linear Unmixing Models

For the linear mixing model, where any observed pixel is a linear combination of the endmembers and its fractional abundances, we have

$$x = Ma + n, \tag{1}$$

where $n \in \mathbb{R}^L$ corresponds to the unfitness of the model (e.g., additive noise). The estimation of the abundances and their corresponding mixing matrix is usually determined by minimizing a cost function of the form:

$$J(A, M) = \|X - MA\|_F^2 + \mu J_{\text{reg}}(A, M), \tag{2}$$

under the non-negativity (ANC) and sum-to-one (ASC) constraints of each a_i, and where $J_{\text{reg}}(\cdot)$ represents some regularization done on both abundances and endmembers, μ being a positive hyperparameter that controls the trade-off between the fitness and the regularization, and $\|\cdot\|_F$ is the Frobenius norm. If the resolution is done by alternating between parameters A and M, such as in non-negative matrix factorization, the estimation of A is often called the Sparse Coding step.

2.2. Nonlinear Unmixing Models

Due to the scattering effects and interactions between endmembers, the linear hypothesis (1) becomes unsuitable. Thus, a more general form would be:

$$x = \phi(Ma) + n, \tag{3}$$

where $\phi : \mathbb{R}^L \to \mathbb{R}^L$ is an unknown nonlinear function to be determined. Two well-known mixing models are often used to imitate the nonlinear behavior, the bilinear [10] and post-nonlinear [11] models, where the nonlinearity ϕ is substituted by a linear trend and an additive nonlinear term, respectively. The bilinear model is defined in [10] as

$$x = Ma + \sum_{i=1}^{R-1} \sum_{j=i+1}^{R} a_i a_j (m_i \odot m_j) + n, \tag{4}$$

and the post-nonlinear model in [11] as

$$x = Ma + Ma \odot Ma + n, \tag{5}$$

where $\odot$ denotes the element-wise product.

In [12], Chen et al. defined the additive nonlinear fluctuation term by a function Φ that depends only on the endmembers, namely

$$x = Ma + \Phi(M) + n, \tag{6}$$

where $\Phi(M)$ is obtained through a kernel method where the closest kernel to the application context is being selected. For example, for a bilinear mixing model, they found the process is closely related to a homogeneous kernel of degree 2. More recently, Zhao et al. proposed in [22] a model where the fluctuation term depends on both the endmembers and the abundances and extracted by means of neural networks, namely

$$x = Ma + \Phi(M, a) + n. \tag{7}$$

This comes to the fact that a material with low abundance should not have a high contribution to the linear or nonlinear component.

2.3. Nonlinear Models Using Autoencoders

Autoencoders are a type of neural network that learns a latent representation of the inputs, with the objective to have an output as close as possible to the input. In the context of hyperspectral unmixing, one seeks to map the spectra to a latent space, namely by learning the abundances as a latent representation.

Autoencoders are based on two main building blocks, an encoder that compresses the input x, namely

$$\Phi^{\text{enc}}(x) = a \tag{8}$$

with $\Phi^{\text{enc}} : \mathbb{R}^L \to \mathbb{R}^R$, and a decoder that reconstructs the input, namely

$$\Phi^{\text{dec}}(a) = \hat{x}, \tag{9}$$

with $\Phi^{\text{dec}} : \mathbb{R}^R \to \mathbb{R}^L$. Each of the building blocks of the autoencoder network can be determined by minimizing the reconstruction error between the input x_i and the output $\hat{x}_i$, namely $x_i \approx \hat{x}_i$.

In the context of HSU, the autoencoder model takes the general form:

$$x \approx \Phi^{\text{dec}}_{\mathcal{M}}(a) \tag{10}$$

where the mixing matrix is derived from parameters $\mathcal{M}$ of the decoder and the abundance estimation is replaced by the encoded part of the network instead of the usual sparse coding step. Figure 1 shows an illustration of hyperspectral unmixing using autoencoders.

In [19], the authors presented an autoencoder network for hyperspectral unmixing. First, by means of fully-connected layers, they map the input spectrum into a latent space of dimension R. This is followed by a batch normalization layer to whiten the data and speed up the learning process. After that, a dynamical soft thresholding layer is used to enforce sparsity in the abundances. Finally, a Gaussian dropout layer is added that acts as a powerful regularizer through white noise injection. For the decoder, the reconstruction is done using a single fully-connected with the weights of this layer are considered as the endmembers. In [18], the authors proposed a new encoder architecture based on 1D convolutional layers with the same linear decoder architecture. The encoder consists of 5 1D convolutional layers, each is followed by a pooling layer. After that, 2 fully-connected layers are used to reach the feature space of dimension R.

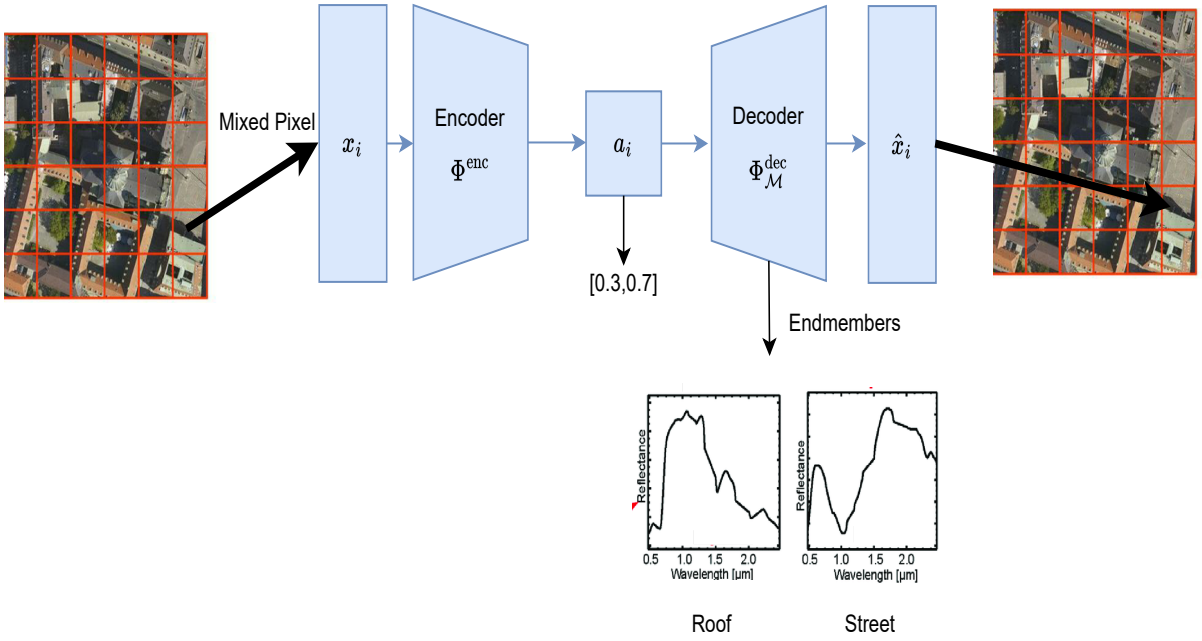


Figure 1. Representation of hyperspectral unmixing using an autoencoder.

In the same manner, Ref. [22] uses a deep encoder architecture with 4 fully-connected layers. However, to address the case of nonlinearity, the decoder starts with a fully-connected layer with $R \times L$ neurons where the weights of this layer is a block diagonal version of the reweighted endmembers matrix. Each endmember is reweighted by its abundances. This is followed by two fully-connected layers to generate the nonlinear interactions. To generate the linear part, a step-wise summation operator is used on the layer with $R \times L$ neurons. The output is the sum of linear and nonlinear parts. An extension of [22] is proposed in [24] to perform spectral–spatial unmixing using a 3D-CNN Autoencoder Network with the same decoder architecture, where the encoder consists of five 3D convolutional layers. In [25], the authors introduced a Autoencoder network based on nonlinear fluctuation over a linear mixture. They impose restrictions on the encoder to be able to invert the mixing process by making use of the pseudoinverse of the endmember matrix, with both encoder and decoder being composed of sequential fully-connected layers. Another set of autoencoder techniques for unmixing is based on adversarial training, such as in [26]. It is based on the assumption that pixels in the same region share the same statistical properties and can be modeled with a prior distribution. It then uses adversarial training to transfer spatial information to the network.

2.4. Proposed Method

In the following, we present the proposed method to perform blind nonlinear unmixing by estimating the abundances and the endmembers through convolutional autoencoders. In our method, we assume that the number of endmembers is known; otherwise, one can estimate it efficiently with the eigengap strategy for example [2].

2.4.1. Proposed Autoencoder Architecture

The proposed method is a pixel-based autoencoder that takes each spectrum, compresses it into a bottleneck layer using an encoder and then tries to reconstruct it back through the decoder. The role of the encoder is to extract hierarchical features that propagate from each hidden layer to another until we reach the latent space. In fully-connected layers, the propagation of information from one layer to another is through a weighted sum of all the inputs in a layer. This makes the training parameters change as a function of the input shape which could affect the convergence of the model. To overcome this issue, we use convolutional layers where the mapping between each hidden layer is

obtained after a convolution operation between the weights of a specific kernel and the input (To manipulate the shape of output of a convolutional layer to reach the latent space dimension, we can adjust the kernel size, stride, padding, and dilation used according to $L_{out} = (L_{in} + 2 \times \text{padding} - \text{dilation} \times (\text{kernel size} - 1) - 1)/\text{stride} + 1)$. In spectral data analysis, the end-to-end CNN approach with local connections and shared weights has proved its efficiency in extracting local features efficiently from a full spectrum [27]. Another way we also used to compress the input and reduce the number of trainable parameters is through pooling layers. Two well-known pooling techniques are often used, average and max pooling. In our case, max pooling is used where the maximum value of a specific window in the feature map is taken.

After obtaining the abundances, a decoder is used to reconstruct the input through a linear and additive nonlinear part. The linear part can be generated through a simple matrix multiplication between the abundances generated and the endmember matrix M. For the nonlinear part, convolutional filters are used to extract neighborhood information at several wavelengths with the ability to discover intricate patterns in high dimensional data, reducing the need for manual effort in preprocessing and feature engineering [28]. We motivate the use of convolutional layers from the fact that, in reflectance spectra, the spectral bands are often highly correlated with the neighboring spectral bands. This can be seen by looking to the correlation matrix of the spectral bands of the hyperspectral dataset Urban shown in Figure 2 (see Section 3.2.2 for more details on the dataset), where the four main blocks around the diagonal illustrate the high correlations between their underlying spectral bands.

In the following, we present in detail the encoder and decoder of the proposed autoencoder, with the intuition behind them. We also formulate the unmixing model provided by our architecture.

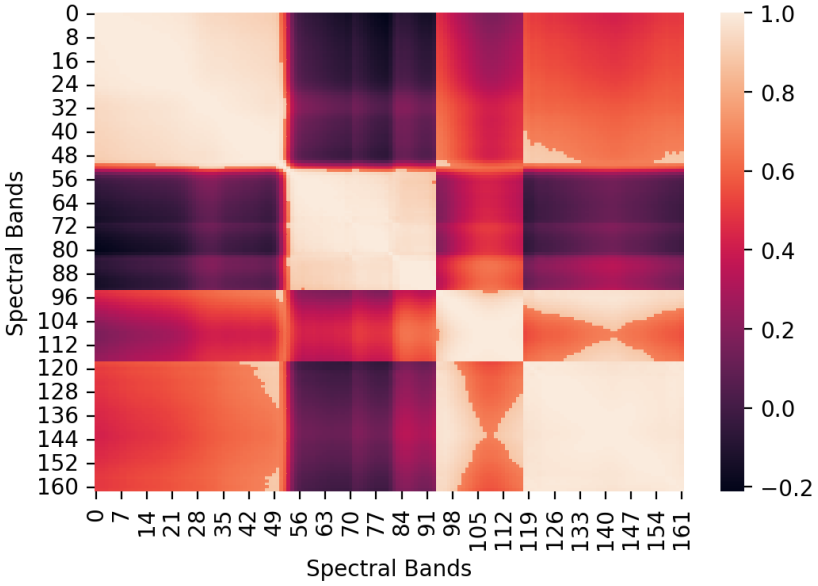


Figure 2. Heatmap of the correlation matrix of the spectral bands of the Urban Dataset.

2.4.2. Encoder

In this part, we benefit from recent advances of CNNs for time series classification where deep features of the input time series are extracted automatically by using convolution and pooling operations [29]. More precisely, we follow the successful implementation of AlexNet in time series classification tasks [30], where the architecture consists of very small filters (5×5) with 5 convolutional layers and 2 fully-connected layers. In our case, 5 convolutional layers and max pooling layers are used. After the last convolution, a fully-connected layer is used to map the extracted features into the abundance vector. The encoding process is illustrated in Figure 3 and the detailed architecture is found in the

upper part of Table 1. After each convolution, nonlinear activation can be used to boost the nonlinearity in the architecture. Several activation functions can be used like Sigmoid, tanh, ELU, Leaky ReLU, or ReLU. In our case, ReLU obtained the best results. The next step is to ensure the nonnegativity and sum-to-one constraints. Inspired from [25], the absolute value rectification is used as in the following equation:

$$a_i = \frac{|a_i|}{\sum_{i=1}^{R} |a_i|}. \tag{11}$$

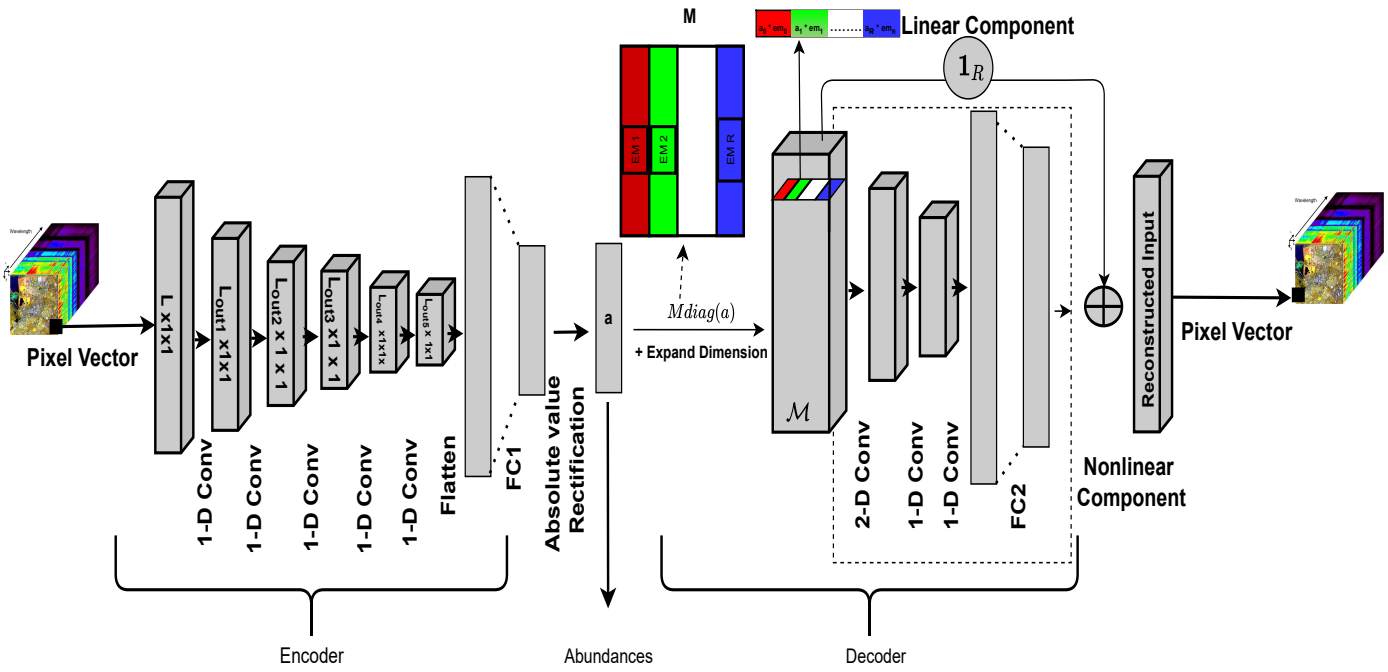


Figure 3. Diagram of the proposed system.

Table 1. Architecture of the proposed network.

	Layers	Number of Filters	Size of Each Filter	Activation Function	Pooling	Output Size
Encoder	1-D Conv	16	1×5	ReLU	Max Pooling 1×2	L_{out1}
	1-D Conv	32	1×5	ReLU	Max Pooling 1×2	L_{out2}
	1-D Conv	64	1×5	ReLU	Max Pooling 1×2	L_{out3}
	1-D Conv	64	1×5	ReLU	Max Pooling 1×2	L_{out4}
	1-D Conv	64	1×5	ReLU	Max Pooling 1×2	L_{out5}
	FC1	1	-	ReLU	-	$R \times 1$
Absolute Value Rectification		-	-	-	-	$R \times 1$
Decoder	Linear	1	-	ReLU	No	$L \times 1$
	2-D Conv	64	(K,R)	ReLU	No	$L_{out6} \times 1 \times 1$
	1-D Conv	64	K	ReLU	No	$L_{out7} \times 1 \times 1$
	1-D Conv	128	K	ReLU	No	$L_{out8} \times 1 \times 1$
	FC2	1	-	ReLU	-	$L \times 1$

2.4.3. Decoder

In the same sense as in (7), we seek a reconstruction model that integrates a linear part and a nonlinear one, namely

$$\begin{aligned} \hat{x} = \Phi_{\mathcal{M}}^{\text{dec}}(a) &= Ma + \Phi(M, a) \\ &= \hat{x}_{\text{linear}} + \hat{x}_{\text{nonlinear}}. \end{aligned} \tag{12}$$

To generate the two parts, linear and nonlinear parts, it is essential to create a new matrix $\mathcal{M} \in \mathbb{R}^{L \times R}$, the re-weighted version of M where each column is multiplied by its abundance. This matrix can be obtained as following:

$$\mathcal{M} = M\text{diag}(a), \qquad \text{with diag}(a) = \begin{bmatrix} a_1 & & \\ & \ddots & \\ & & a_R \end{bmatrix}, \tag{13}$$

where $\text{diag}(a)$ is the diagonal matrix with vector a in the diagonal. Thus, the unmixing model provided by the proposed architecture is defined by

$$\hat{\mathrm{x}} = Ma + \Phi(M\text{diag}(a)). \tag{14}$$

The linear part is obtained by the summation of the matrix $\mathcal{M}$ along its columns as following:

$$\hat{\mathrm{x}}_{\text{linear}} = \mathcal{M}\, \mathbf{1}_R \tag{15}$$

For the nonlinear part, we use the matrix $\mathcal{M}$ also to estimate the nonlinear function Φ that represents the interactions between different endmembers. To estimate Φ, our process starts with a 2D convolutional layer to the input $\mathcal{M}$. The 2D kernels help in learning the nonlinearity by extracting features across wavelength and endmember domain. Therefore, the nonlinear part $\hat{x}_{\text{nonlinear}}$ is obtained by stacking four layers as following:

$$\begin{aligned} \hat{x}_{\text{nonlinear}_1}[i] &= \sum_{v=-R/2}^{R/2} \sum_{u=-K/2}^{K/2} W_1[u, v]\mathcal{M}[i-u, v] \\ \hat{x}_{\text{nonlinear}_2}[i] &= \sum_{u=-K/2}^{K/2} W_2[u]\hat{x}_{\text{nonlinear}_1}[i-u] \\ \hat{x}_{\text{nonlinear}_3}[i] &= \sum_{u=-K/2}^{K/2} W_3[u]\hat{x}_{\text{nonlinear}_2}[i-u] \\ \hat{x}_{\text{nonlinear}} &= W_{FC2} \odot \hat{x}_{\text{nonlinear}_3} \end{aligned} \tag{16}$$

where W_1 is a 2D kernel (u and v being the indices in the kernel matrices and i is the row index of $\mathcal{M}$), W_2 and W_3 are 1D kernels while W_{FC2} is the weight matrix of the last fully-connected layer (denoted by $FC2$ in Figure 3). For the kernel size across the endmember domain, it was found that kernels of width R obtain the best results. However, for selecting the kernel size K across the wavelength domain, the process depends on the application, i.e., the degree of nonlinearity. For datasets with a moderate level of nonlinearity, a small kernel can be used. However, for high degrees of nonlinearities, increasing the kernel size is necessary. It is always a trade-off between a valid receptive field and the number of parameters [31]. Deeper architectures can extract more abstract features [32], therefore two 1D convolutions with the same kernel size K are added and followed by 1 fully-connected layer to piece these features into the reconstructed input. The architecture of the decoder is resumed in the lower part of Table 1.

2.5. Objective Function

Our loss function is composed of the mean-square error between the input and the output of the autoencoder with additional regularization needed to control the energy of nonlinearity. Thus, L_2-norm of the weights W_{FC2} is minimized to avoid large scale nonlinear components compared to the linear component. Besides, as the output spectrum comes as a function of the endmembers according to Equation (14), the nonlinear fluctuation term in the output introduces a fluctuation effect on the endmembers too. To maintain the endmembers smooth without fluctuations, a smoothing regularization is needed to minimize the total variation (TV) in each column of the endmember matrix. Therefore, the loss function $\mathcal{L}$ is

$$\begin{aligned}\mathcal{L}(x,\hat{x}) =& \mathcal{L}_{\text{mse}} + \mathcal{L}_{\text{nonlinear}} + \mathcal{L}_{\text{TV}} \\ =& \frac{1}{N}\sum_{i=1}^{N}\|x_i - \hat{x}_i\|^2 + \lambda_{\text{nl}}\|W_{FC2}\|^2 + \lambda_{\text{TV}}\sum_{i=1}^{R}\sum_{j=1}^{L-1}\left|[m_i]_{j+1} - [m_i]_j\right|,\end{aligned} \tag{17}$$

where λ_{nl} and λ_{TV} are coefficients that control the regularization degree. The Adam optimizer [33] is used to minimize the loss function through a mini-batch gradient descent process. During backpropagation, the endmembers matrix M is updated once the matrix $\mathcal{M}$ is updated. To ensure the nonnegativity on M, clamping for negative values is used.

3. Experimental Results

In this section, the proposed architecture is tested and compared to several state-of-the-art unmixing methods. The testing was done using synthetic data as well as real airborne image data. For the evaluation of abundance estimation, we use the root mean square error (RMSE), namely

$$\text{RMSE} = \sqrt{\frac{1}{NR}\sum_{i=1}^{N}\|a_i - \hat{a}_i\|^2}, \tag{18}$$

where a_i and $\hat{a}_i$ denote the true and estimated abundance vectors of the i-th spectrum. For endmember estimation, we use the spectral angle distance (SAD), namely

$$\text{SAD} = \cos^{-1}\left(\frac{m^\top \hat{m}}{\|m\|\,\|\hat{m}\|}\right), \tag{19}$$

where m and $\hat{m}$ represent the true and estimated endmembers, and the spectral information divergence (SID), namely

$$\text{SID}(m \mid \hat{m}) = \sum_j p_j \log\left(\frac{p_j}{\hat{p}_j}\right), \tag{20}$$

with $p = m/1^\top m$, $\hat{p} = \hat{m}/1^\top \hat{m}$ and p_j is the j-th entry of p.

Table 2 summarizes the benchmark methods used for comparison. In the first 2 methods, VCA and N-FINDR assume that the spectra are linearly mixed, thus the endmember extraction step is considered linear while the abundance estimation takes into consideration the nonlinear interactions between these endmembers. For Pixel-based DCAE in [18], the authors used spectral information divergence loss (SID) and in a supervised strategy. As we encountered bad reconstruction errors, and for comparison to be fair (architecture wise), we added a modified version that uses MSE loss and we let the endmembers to be updated during training. For r-NMF, DAEN, Pixel-based DCAE, Modified Pixel-based DCAE as well as the proposed method, the same initialization was done using N-FINDR.

Table 2. Benchmark methods used for comparison.

Method	Description
VCA & K-Hype	Vertex Component Analysis is a classic geometric method and K-Hype is a linear/nonlinear fluctuation model that approximates the nonlinearity with the kernel trick [12].
N-FINDR & NDU	N-FINDR is a classic endmember extraction method and NDU performs a nonlinear abundance estimation using neighborhood information [34].
r-NMF	Robust non-negative matrix factorization performs robust nonlinear endmember and abundance estimation via block-coordinate descent algorithm [35].
HyperspecAE	Hyperspectral Linear Unmixing Using A Neural Network Autoencoder [19].
DAEN	Deep autoencoder network for linear/nonlinear fluctuation unmixing [22].
CYCUNET	Cycle-consistency Unmixing Network by Learning Cascaded Autoencoders [36].
Pixel-based DCAE	Deep Convolutional Autoencoder for Linear Hyperspectral Unmixing [18].
Modified Pixel-based DCAE	Deep Convolutional Autoencoder for Linear Hyperspectral Unmixing [18] trained with MSE loss function and in an unsupervised way.

3.1. Experiments with Synthetic Data

For the synthetic data, four endmembers were selected from the United States Geological Survey (USGS) digital spectral library [37]. Each endmember is composed of 224 contiguous bands. The abundances were generated from a Dirichlet distribution and a total of 100×100 pixels were generated. Additive zero-mean Gaussian noise is added to the data with a Signal-to-Noise Ratio (SNR) of 20, 30 and 40 dB. Three mixtures were made to evaluate the performance of the methods based on the type of nonlinearity. The used mixing models are linear, bilinear, and post-nonlinear models.

The network architecture is shown in Table 1. For the number of filters per convolutional layer, inspired by AlexNet architecture, we start with 16 filters then 32 and then 3 layers with 64 filters. For the decoder, more filters were needed to estimate the nonlinearity; therefore, two layers with 64 filters were used followed by a layer with 128 filters. For setting the kernel size K, small filters were used of size 5 for the linear model. However, for the bilinear and post-nonlinear models, where the nonlinearities add large fluctuations to the output, better results were achieved by filters of size 40. The learning rate was set to 10^{-3} and the batch size was set to 32. In setting a batch size there is a trade off between convergence and generalization and the choice depends on the data itself. In our case, our CNN behaved better with small batch sizes. The model then was trained for 100 epochs. For the regularization parameters, manual tuning was done and λ_{nl} was set to 10^{-3} and λ_{TV} to 10^{-6}.

In Table 3, the abundance RMSE with the ground truth is shown according to different models and SNR values. The model was run 5 times and then the average RMSE as well as the standard deviation was recorded. Tables 4 and 5 also evaluate the performance of the compared methods in endmember's estimation according to the mean of SAD and SID metrics on the four extracted endmembers. The deep encoder architecture trained with MSE loss leads to achieving robust abundance RMSE scores as well as endmember estimations as function of degree of nonlinearities. This can be useful for strongly mixed scenes captured with noisy imaging devices of low SNRs. By looking at the abundance results, it is clear that the proposed model has a superior performance in all mixing models, except for the linear ones with SNR at 30 and 40 dB. SID is known to deliver better abundances, but with output spectra scale problems [19]. However, in case of synthetic data, the endmembers generated by N-FINDR and given to the Pixel-based DCAE were close to the true endmembers by looking to Tables 4 and 5. These endmembers were also fixed throughout the whole process. Thus the output spectra of the Pixel-based DCAE trained by SID can be of appropriate scale and therefore with better abundances. This advantage no longer holds in case of low SNR and nonlinear effects. Other state-of-the-art methods show a close MSE score in linear

models while in the bilinear and PNMM models an obvious improvement is obtained. For rNMF, a block descent algorithm might fail to converge although being initialized with the same endmembers as in the proposed method. Additionally, the proposed method got the lowest SAD and SID scores in almost all the models. SID score can be more affected than SAD by the TV regularization introduced for the reason that SID focuses more on the spectral variabilities by computing the discrepancy between the probability distributions produced by the spectral signatures. However, the obtained results are still sufficient especially in nonlinear models. Figure 4 shows a comparison between true and extracted endmembers, i.e., the endmember matrix M, in a bilinear model case with 20 dB SNR.

Table 3. Abundance RMSE comparison of the synthetic data (best results are highlighted in red).

	SNR = 20 dB			**SNR = 30 dB**		
	Linear	**Bilinear**	**PNMM**	**Linear**	**Bilinear**	**PNMM**
NFINDR + NDU	0.0632 ± 0.0045	0.1073 ± 0.0023	0.0801 ± 0.0034	0.0593 ± 0.0074	0.0889 ± 0.0033	0.0794 ± 0.0062
VCA + K-Hype	0.1626 ± 0.0185	0.1634 ± 0.0077	0.1607 ± 0.0186	0.0674 ± 0.0050	0.0782 ± 0.0033	0.1113 ± 0.0097
rNMF	0.0808 ± 0.0107	0.0871 ± 0.0041	0.1127 ± 0.0029	0.0824 ± 0.0062	0.0859 ± 0.0050	0.1142 ± 0.0014
HyperspecAE	0.2848 ± 0.0122	0.2848 ± 0.0123	0.2848 ± 0.0126	0.2848 ± 0.0131	0.2849 ± 0.0135	0.2848 ± 0.0138
DAEN	0.0665 ± 0.0100	0.0782 ± 0.0020	0.0894 ± 0.0067	0.0679 ± 0.0062	0.0678 ± 0.0045	0.0941 ± 0.0040
CYCUNET	0.2662 ± 0.0124	0.2846 ± 0.0205	0.3574 ± 0.0231	0.3950 ± 0.0156	0.4228 ± 0.0189	0.3823 ± 0.0164
Pixel-based DCAE	0.1255 ± 0.0020	0.1407 ± 0.0120	0.1431 ± 0.0220	**0.0563 ± 0.0042**	0.0611 ± 0.0240	0.1216 ± 0.0111
Modified Pixel-based DCAE	0.2449 ± 0.0112	0.0999 ± 0.0116	0.0967 ± 0.0107	0.2554 ± 0.0072	0.2684 ± 0.0104	0.1126 ± 0.0108
Proposed method	**0.0571 ± 0.0060**	**0.0578 ± 0.0051**	**0.0614 ± 0.0055**	0.0578 ± 0.0064	**0.0578 ± 0.0055**	**0.0586 ± 0.0042**

	SNR = 40 dB		
	Linear	**Bilinear**	**PNMM**
NFINDR + NDU	0.0577 ± 0.0082	0.0907 ± 0.0026	0.0659 ± 0.0070
VCA + K-Hype	0.0583 ± 0.0029	0.0758 ± 0.0027	0.0792 ± 0.0104
rNMF	0.0871 ± 0.0089	0.0942 ± 0.0059	0.1155 ± 0.0009
HyperspecAE	0.2848 ± 0.0141	0.2849 ± 0.0145	0.2848 ± 0.0138
DAEN	0.0608 ± 0.0067	0.0695 ± 0.0065	0.0896 ± 0.0042
CYCUNET	0.3562 ± 0.0185	0.3562 ± 0.0241	0.4044 ± 0.0217
Pixel-based DCAE	**0.0463 ± 0.0630**	0.0618 ± 0.0840	0.1159 ± 0.0060
Modified Pixel-based DCAE	0.1134 ± 0.0073	0.1037 ± 0.0068	0.1009 ± 0.0103
Proposed method	0.0595 ± 0.0058	**0.0577 ± 0.0058**	**0.0620 ± 0.0029**

Table 4. Endmember SAD comparison of the synthetic data (best results are highlighted in red).

	SNR = 20 dB			**SNR = 30 dB**			**SNR = 40 dB**		
	Linear	**Bilinear**	**PNMM**	**Linear**	**Bilinear**	**PNMM**	**Linear**	**Bilinear**	**PNMM**
NFINDR + NDU/ Pixel-based DCAE	0.0998	0.1343	0.1675	0.0652	0.0856	0.1544	0.0621	0.0842	0.1534
VCA + K-Hype	0.0645	0.0929	0.0991	0.0531	0.0558	0.1066	0.0532	0.0586	0.0951
rNMF	0.0829	0.1016	0.1402	0.0853	0.1081	0.1345	0.1087	0.1098	0.1334
HyperspecAE	0.0989	0.1103	0.1789	0.0852	0.1185	0.1544	0.8215	0.1424	0.1534
DAEN	0.0750	0.0788	**0.0904**	0.0751	0.1108	0.1014	0.0823	0.1208	0.1133
CYCUNET	0.1197	0.1259	0.1547	0.0513	0.0761	0.0894	0.03723	0.0679	0.0758
Modified Pixel-based DCAE	0.2537	0.1060	0.1340	0.2596	0.3347	0.1450	0.1255	0.1032	0.1319
Proposed method	**0.0423**	**0.0577**	0.1193	**0.0305**	**0.0380**	**0.0905**	**0.0294**	**0.0397**	**0.0919**

Table 5. Endmember SID comparison of the synthetic data (best results are highlighted in red).

	SNR = 20 dB			SNR = 30 dB			SNR = 40 dB		
	Linear	**Bilinear**	**PNMM**	**Linear**	**Bilinear**	**PNMM**	**Linear**	**Bilinear**	**PNMM**
NFINDR + NDU/ Pixel-based DCAE	0.0421	0.0589	0.0641	0.0080	0.0116	0.0189	0.0035	0.0070	0.0116
VCA + K-Hype	0.0405	0.0226	**0.0109**	0.0036	**0.0029**	0.2405	0.0051	**0.0047**	0.0133
rNMF	0.0124	0.0128	0.0317	0.0146	0.0140	0.0290	0.0192	0.0146	0.0288
HyperspecAE	0.0625	0.0432	0.0789	0.0765	0.0125	0.0256	0.0081	0.0450	0.0480
DAEN	0.0142	0.0146	0.0272	0.0095	0.0322	0.0546	0.0091	0.0350	0.0470
CYCUNET	0.0411	0.0434	0.064	0.0079	0.0115	0.0189	0.0034	0.007	0.0115
Modified Pixel-based DCAE	0.1201	0.1172	0.0794	0.0793	0.1847	0.1532	0.1793	0.1224	0.1188
Proposed method	**0.0113**	**0.0095**	0.0233	**0.0033**	0.0047	**0.0051**	**0.0034**	0.0058	**0.0055**

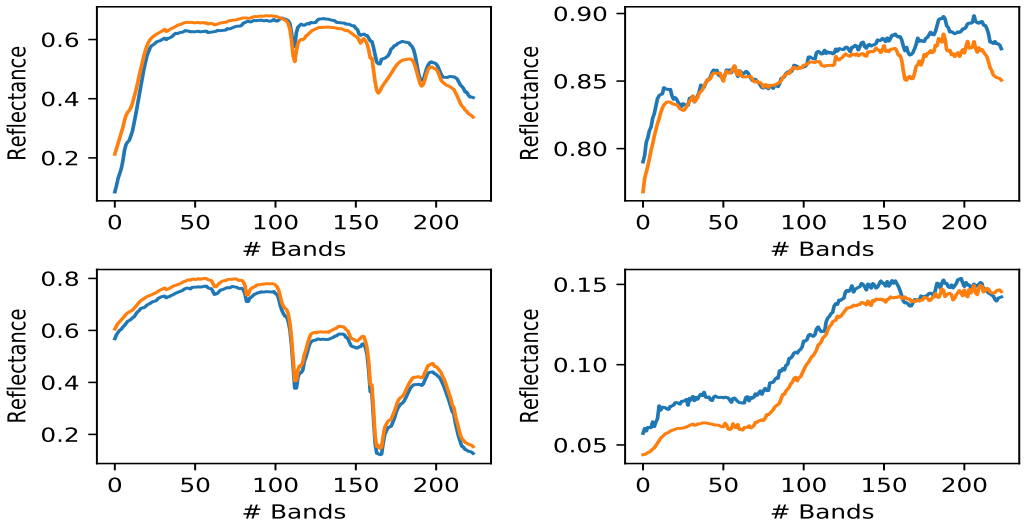
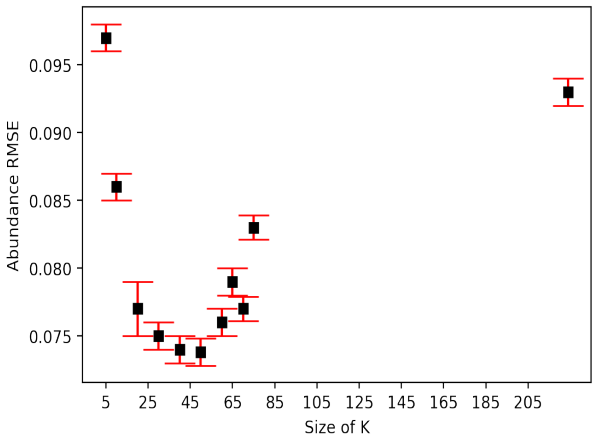


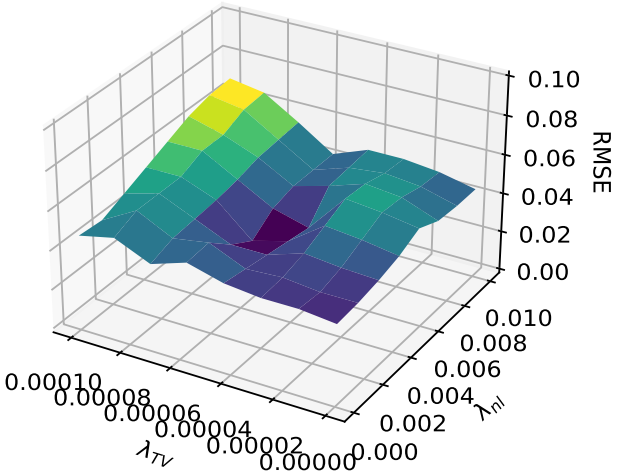
Figure 4. Comparison between extracted endmembers and the four ground truth synthetic endmembers. Orange curves: true endmembers; Blue curves: extracted endmembers (SNR = 20 dB, bilinear model case).

3.1.1. Model Sensitivity with Regard to Kernel Size and Regularization Parameters

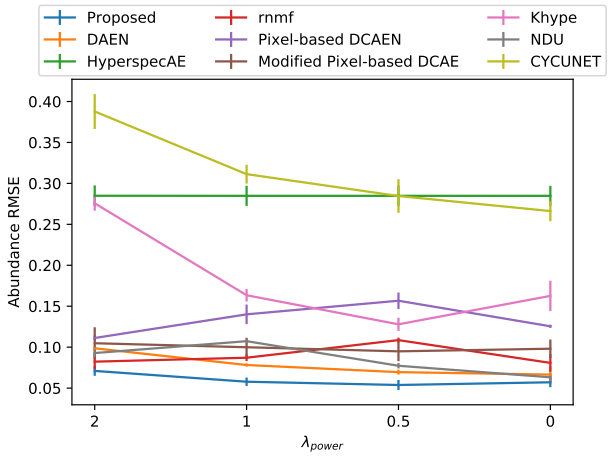
To understand the effect of choosing the kernel size K of the decoder, Figure 5a shows the change of abundance RMSE as function of K. The study was done on the bilinear mixture with SNR = 20 dB. We can clearly see that very small and very large kernel sizes lead to bad RSME values. Besides, to highlight the advantage of convolutional layer over being fully-connected in the decoder, we also studied the extreme case where the kernel size is equal to the number of spectral bands (namely, 224), which means that the convolutional layers behave as a fully-connected one. We can see the RSME is much greater compared to results obtained with a convolutional layer of adequate kernel size. In the same manner, Figure 5b is a a surface plot showing the sensitivity (performance) of the proposed method with respect to the regularization parameters λ_{nl} and λ_{TV}. We can see that around optimal values, the model has low sensitivity to the regularization parameters meaning that it can behave well for a wide range of values of these hyperparameters.



(**a**)



(**b**)



(**c**)

Figure 5. Model performance as function of regularization parameters and kernel size. (**a**) Abundance RMSE vs. kernel size K of the proposed model. (**b**) Abundance RMSE vs. regularization parameters λ_{nl} and λ_{TV}. (**c**) Abundance RMSE vs. regularization parameter λ_{power}.

3.1.2. Model Sensitivity with Regard to Degree of Nonlinearity

To measure the performance of the proposed model as a function of degree of nonlinearity in a dataset, we used synthetic data generated using the bilinear mixture at 20 dB

SNR. To control the degree of the nonlinear term added to the linear one, we introduced a hyperparameter λ_{power} so that the mixture is:

$$x = Ma + \lambda_{power} \sum_{i=1}^{R-1} \sum_{j=i+1}^{R} a_i a_j (m_i \odot m_j) + n. \quad (21)$$

We chose four values for $\lambda_{power} \in \{0, 0.5, 1, 2\}$. The kernel size K is set to 40 as before. The results are shown in Figure 5c. As we can see, the proposed model has a robust and stable performance as the degree of nonlinearity in the dataset varies. This signifies the ability of the model to be used in several environmental settings where the nonlinearity degree varies constantly.

3.2. Experiments with Real Airborne Data

3.2.1. Jasper Ridge Dataset

The Jasper Ridge dataset (https://rslab.ut.ac.ir/data, accessed on 1 March 2022) is a hyperspectral image captured by AVARIS sensor. It originally has 512 × 614 pixels and 224 spectral bands from 380 to 2500 nm. Noisy bands were removed and 198 were selected. A subimage of size 100 × 100 pixels was used, containing four endmembers: Tree, Water, Soil and Road.

The same encoder architecture was used as for the synthetic dataset. However, for the decoder, the kernel size K was set to 5. The batch size was set to 512, learning rate to 10^{-4} and number of epochs to 100. For the regularization parameters, λ_{nl} was set to 10^{-3} and λ_{TV} to 10^{-6}. Figure 6 shows the extracted four endmembers, i.e., the endmember matrix M after training. Figure 7 shows the abundance maps provided by the different state-of-the-art methods as well as the proposed method, except HyperspecAE, which failed to converge to comprehensive abundance maps. These abundance maps are the output of the encoder part. Each row contains the terrain features that belong to one of the four pure materials presented in the scene (the four endmembers of Figure 6). For example, for the terrain zones full of trees (second row), all state-of-the-art methods managed to highlight the same zones. This comes from the fact that trees endmember doesn't cause confusion with other materials. However, the road endmember is well known to be the hardest to estimate its abundance as it is the least represented endmember in the dataset. By looking to the proposed model results, we can see how the model reduces the confusion between road and soil endmember compared to other models. Besides, Figure 8 shows the energy of nonlinear components of nonlinear methods only. These plots can be obtained taking the nonlinear term only of the decoder part. In these plots, we can see how the proposed model assigns higher energy to the road terrain zones to help in a better reconstruction. For Pixel-based DCAE, the SID loss function was able to create more clear abundances for other endmembers. This can be seen by looking to water and trees abundance maps where no confusion between each other compared to the results of other state-of-the-art methods. However, this comes at the expanse of the reconstruction error, since the SID loss is scale invariant, which lead to incorrect scale of output spectrum thus worse reconstruction. The reconstruction error maps for the compared methods are shown in Figure 9. From the reconstruction plots, we can see how the proposed model managed to obtain the best reconstruction for all the terrain features. In the contrary, we can see that Pixel-based achieved the worst reconstruction for almost all the terrain. Table 6 provides the reconstruction error obtained by all the compared models, where the RE metric is defined as follows:

$$RE = \sqrt{\frac{1}{NL} \sum_{i=1}^{N} \|x_i - \hat{x}_i\|_2^2} \quad (22)$$

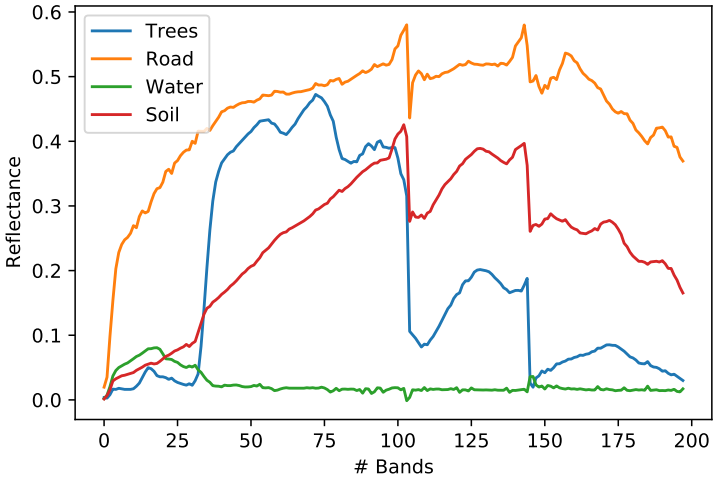


Figure 6. Endmembers extracted by the proposed method for the Jasper Ridge.

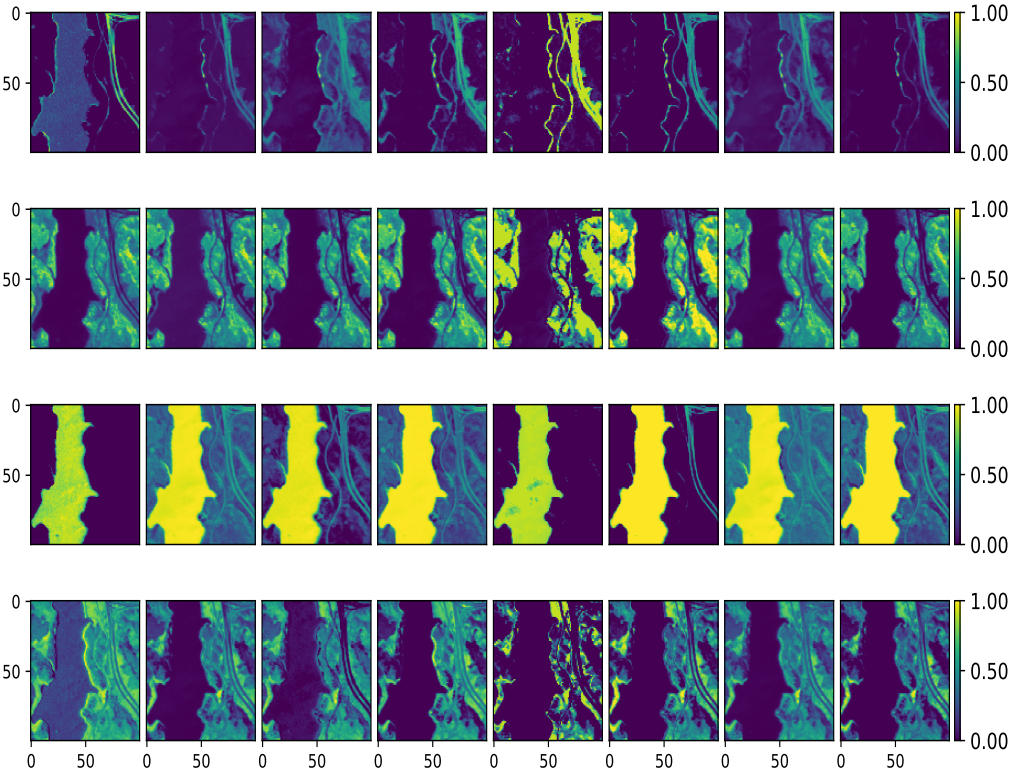


Figure 7. Estimated abundance maps of Jasper Ridge dataset. From left to right: VCA-K-Hype, N-FINDR-NDU, rNMF, DAEN, CYCUNET, Pixel-based DCAE, Modified Pixel-based DCAE, and the proposed method, respectively. From top to bottom: Road, Trees, Water, and Soil, respectively.

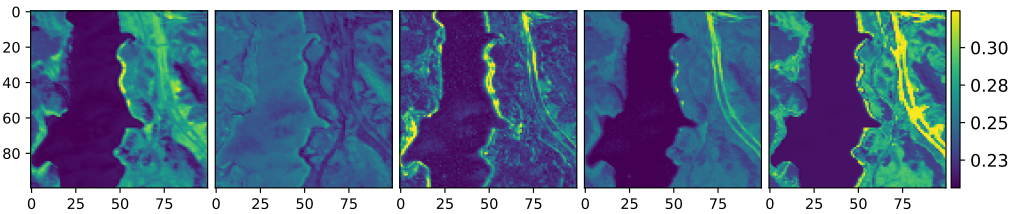


Figure 8. Energy of the nonlinear components of the Jasper Ridge data. From left to right: VCA-K-Hype, N-FINDR-NDU, rNMF, DAEN, and the proposed method, respectively.

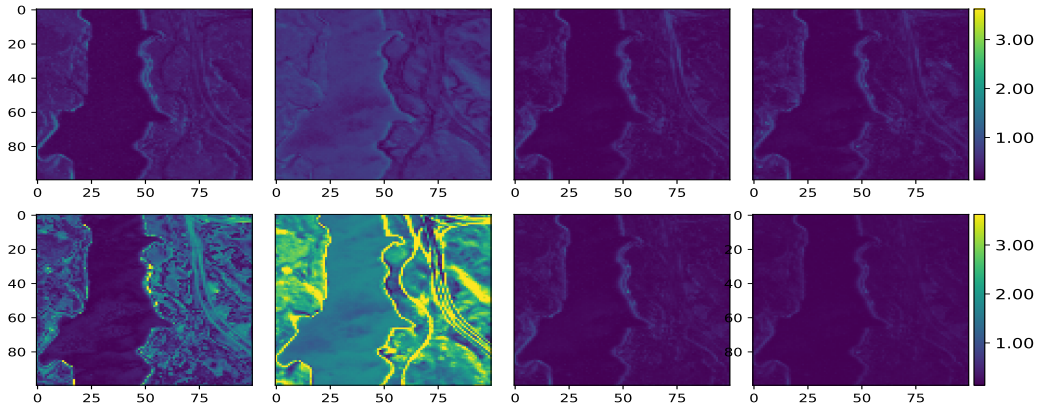


Figure 9. Maps of reconstruction error of the Jasper Ridge data. From upper left to right: VCA-K-Hype, N-FINDR-NDU, rNMF, DAEN. From bottom left to right: CYCUNET, Pixel-based DCAE, Modified Pixel-based DCAE, and the proposed method, respectively.

Table 6. Re-comparison of the real airborne data (best results are highlighted in red).

	RE	
Method	**Jasper Ridge**	**Urban**
KHYPE	0.0151 ± 0.0009	0.0115 ± 0.0002
NDU	0.0185 ± 0.0009	0.0376 ± 0.0003
rNMF	0.0131 ± 0.0004	0.0120 ± 0.0003
HyperspecAE	0.0877 ± 0.0052	0.0702 ± 0.0013
DAEN	0.01 ± 0.0009	0.0059 ± 0.0005
CYCUNET	0.0303 ± 0.0007	0.0764 ± 0.0007
Pixel-based DCAE	0.0771 ± 0.0009	0.229 ± 0.0012
Modified Pixel-based DCAE	0.0113 ± 0.0003	0.0113 ± 0.0003
3-D-CNN Autoencoder Network (results from [24])	0.0118	0.0116
Proposed method	**0.0068 ± 0.0002**	**0.0096 ± 0.0002**

The reconstruction error plots is obtained by subtracting the output of the decoder from the original input. For 3D-CNN [24], due to unavailability of code, we used the results from the author's paper obtained on the same experimental settings as our paper. This includes reconstruction error scores (same formula as in Equation (22)) on real airborne data.

3.2.2. Urban Dataset

The Urban dataset is another well-known hyperspectral dataset used for unmixing with a spatial dimension of 307 × 307 pixels and spectral resolution of 210 wavelengths from 400 to 2500 nm. After removing water absorption zones, only 162 channels are left. This dataset is formed of four endmembers: Asphalt, Dirt, Tree, and Roof.

We used the same architecture as for the Jasper Ridge dataset with a batch size of 512 and trained for 100 epochs. The learning rate was set to 10^{-4}, λ_{nl} to 10^{-3} and λ_{TV} to 10^{-6}. The extracted endmembers are shown in Figure 10. The abundance maps are provided in Figure 11 for all methods except HyperspecAE. Each row contains the terrain features that belong to one of the four pure materials in Figure 10. For example, the first row contains the zones related to asphalt. We can see that most of the compared methods extracted the same terrain features. However, the challenging task in this dataset is to distinguish between dirt and trees zones. We can see that the proposed model achieved the best distinguish with less confusion. NDU+N-FINDR achieved a close performance; however, this relies on the N-FINDR intialization for endmembers that once can be close to optimal and once can be too far, thus providing non-stable performances.The nonlinear maps for nonlinear methods are give in Figure 12. We can see that all the compared methods and the proposed one add higher nonlinear energy to the roof terrain features, which means it is the one with most nonlinear interactions with other endmembers. Once again, the SID leads to clearer

and more sparse abundances, but worse reconstruction errors. By checking Figure 13 and Table 6, we can see that the proposed method got the lowest reconstruction error with respect to all terrain zones.

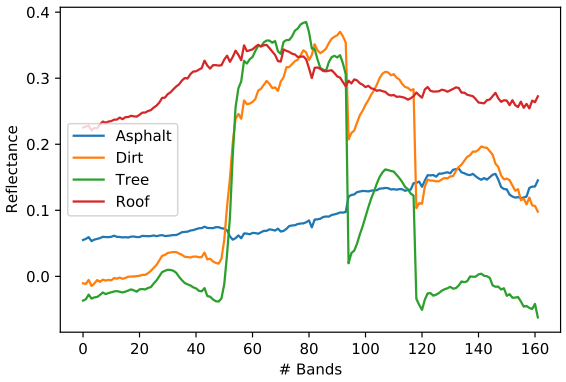


Figure 10. Urban dataset endmembers.

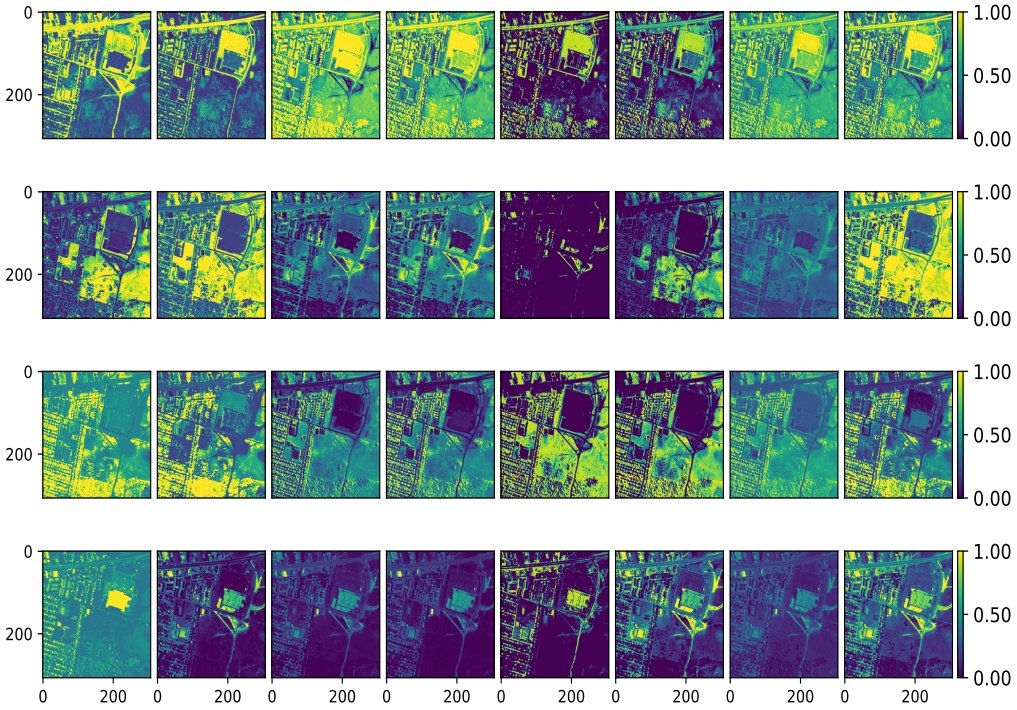


Figure 11. Estimated abundance maps of Urban dataset. From left to right: VCA-K-Hype, N-FINDR-NDU, rNMF, DAEN, CYCUNET, Pixel-based DCAE, Modified Pixel-based DCAE and the proposed method, respectively. From top to bottom: Asphalt, Dirt,Tree, and Roof, respectively.

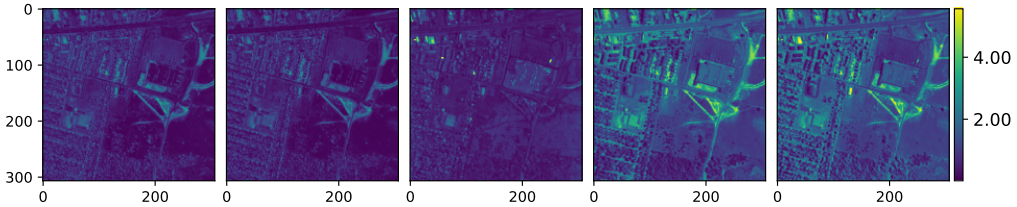


Figure 12. Energy of the nonlinear components of the Urban data. From left to right: VCA-K-Hype, N-FINDR-NDU, rNMF, DAEN and the proposed method, respectively.

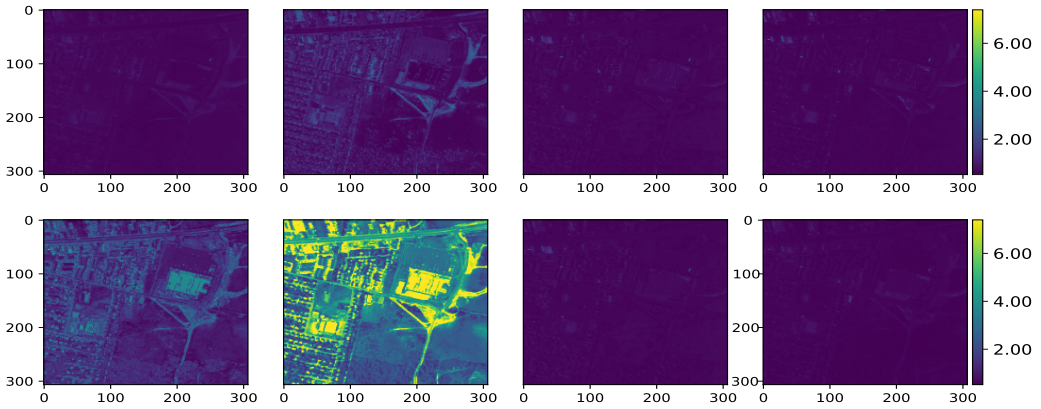


Figure 13. Maps of reconstruction error of the Urban data. From upper left to right: VCA-K-Hype, N-FINDR-NDU, rNMF, DAEN. From bottom left to right: CYCUNET, Pixel-based DCAE, Modified Pixel-based DCAE, and the proposed method, respectively.

3.3. Computation Time

In this section, we evaluate the performance of the proposed method in terms of computation time. All the experiments were carried on the same CPU device (Intel Xeon E5-2643 v2) and same GPU (NVIDIA GTX 1060). The average computation time over 5 runs done on each dataset is recorded in Table 7. From the results, we can see that the proposed method achieves similar computational complexity with state-of-the-art deep autoencoders (DAEN, Pixel-based DCAE and HyperspecAE). Autoencoders achieve better unmixing scores while with relatively longer training time. However, training time of neural networks can be further optimized using parallel GPU training and using a portion of data as a training set.

Table 7. The average computation time for each dataset.

Method	**Time [Sec]**		
	Synthetic	**Jasper**	**Urban**
NDU	1410	1380	7840
K-Hype	19	25	190
rNMF	90	95	170
HyperspecAE	450	330	950
DAEN	670	492	1426
Cycunet	50	52	380
Pixel-based DCAE	590	433	1255
Modified Pixel-based DCAE	600	441	1279
Proposed method	650	477	1383

4. Conclusions

In this paper, we presented an end-to-end convolutional autoencoder for nonlinear hyperspectral unmixing. It is based on a linear mixture with an additive nonlinear fluctuation mixture. Our intuition is to take advantage of the capability of CNNs in extracting deep features and solving complex problem in computer vision tasks. To this end, the proposed autoencoder used a convolutional encoder to boost the performance of compressing the input spectrum into a narrow subspace (abundances) and a convolutional decoder that learns deeply the interactions between the endmembers. To sum up the experimental results, the proposed method was tested on both synthetic and well-known real datasets, with superior performance in almost all scenarios compared to state-of-the-art nonlinear unmixing methods. In real-world scenarios, hyperspectral images captured for large terrain landscapes can suffer from the mix indistinguishable materials such as dirt and trees. The performance of the proposed model proved to overcome this challenge by achieving

the lowest confusion possible. In addition, the capability of working under low SNRs, make the proposed model more adequate to real scenes where weather conditions can affect greatly the imaging sensor and the amount of noise in the data. Future work will consist in three aspects. The first is managing to add the SID loss to the total loss function used for training with adequate hyperparameter that controls its energy and manage to get both better abundances and reconstruction error. The second is integrating contextual information to the unmixing process, which would allow us to get better results; the price to pay is the computational complexity. Finally, improving the performance by resolving the issue of endmember variability.

Author Contributions: Conceptualization, M.D.; methodology, M.D.; software, M.D.; validation, M.B., P.H. and A.V.E.; formal analysis, M.D.; investigation, M.B. and P.H.; resources, M.D.; data curation, M.D.; writing—original draft preparation, M.D.; writing–review and editing, M.B. and P.H.; visualization, M.D.; supervision, M.B. and P.H.; project administration, M.B. and P.H.; funding acquisition, A.V.E. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the ANRT (Agence Nationale Recherche Technologie, French Association for Research and Technology) and TELLUX Company.

Data Availability Statement: Data available upon reasonable request to the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

List of mathematical symbols and acronyms:

Symbol	Definition
X	Hyperspectral image $\in \mathbb{R}_{+}^{L\times N}$
M	Endmember matrix $\in \mathbb{R}^{L\times R}$
A	Abundance vectors $\in \mathbb{R}_{+}^{R\times N}$
Φ	Nonlinear mixing function
$\mathcal{M}$	$\in \mathbb{R}^{L\times R}$ the re-weighted version of M
W_{FC2}	Weights of last fully connected layer
L	Loss function
ANC	Abundance Non-negativity Constraint
ASC	Abundance Sum-to-one Constraint
N	Number of pixels
R	Number of endmembers
L	Number of spectral bands
SID	Spectral Information Divergence
SAD	Spectral Angle Distance
SNR	Signal-to-Noise Ratio

References

1. Keshava, N.; Mustard, J.F. Spectral unmixing. *IEEE Signal Process. Mag.* **2002**, *19*, 44–57. [CrossRef]
2. Halimi, A.; Honeine, P.; Kharouf, M.; Richard, C.; Tourneret, J.Y. Estimating the Intrinsic Dimension of Hyperspectral Images Using a Noise-Whitened Eigengap Approach. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 3811–3821. [CrossRef]
3. Winter, M.E. N-FINDR: An algorithm for fast autonomous spectral end-member determination in hyperspectral data. *Imaging Spectrom. Int. Soc. Opt. Photonics* **1999**, *3753*, 266–275.
4. Nascimento, J.M.; Dias, J.M. Vertex component analysis: A fast algorithm to unmix hyperspectral data. *IEEE Trans. Geosci. Remote Sens.* **2005**, *43*, 898–910. [CrossRef]
5. Heinz, D.C. Fully constrained least squares linear spectral mixture analysis method for material quantification in hyperspectral imagery. *IEEE Trans. Geosci. Remote Sens.* **2001**, *39*, 529–545. [CrossRef]
6. Bioucas-Dias, J.M.; Figueiredo, M.A. Alternating direction algorithms for constrained sparse regression: Application to hyperspectral unmixing. In Proceedings of the 2010 2nd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing, Reykjavik, Iceland, 14–16 June 2010; pp. 1–4.

7. Miao, L.; Qi, H. Endmember extraction from highly mixed data using minimum volume constrained nonnegative matrix factorization. *IEEE Trans. Geosci. Remote Sens.* **2007**, *45*, 765. [CrossRef]
8. Honeine, P.; Richard, C. Geometric unmixing of large hyperspectral images: A barycentric coordinate approach. *IEEE Trans. Geosci. Remote Sens.* **2012**, *50*, 2185–2195. [CrossRef]
9. Dobigeon, N.; Tourneret, J.Y.; Richard, C.; Bermudez, J.C.M.; McLaughlin, S.; Hero, A.O. Nonlinear unmixing of hyperspectral images: Models and algorithms. *IEEE Signal Process. Mag.* **2013**, *31*, 82–94. [CrossRef]
10. Halimi, A.; Altmann, Y.; Dobigeon, N.; Tourneret, J.Y. Nonlinear unmixing of hyperspectral images using a generalized bilinear model. *IEEE Trans. Geosci. Remote Sens.* **2011**, *49*, 4153–4162. [CrossRef]
11. Altmann, Y.; Halimi, A.; Dobigeon, N.; Tourneret, J.Y. Supervised nonlinear spectral unmixing using a polynomial post nonlinear model for hyperspectral imagery. In Proceedings of the 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Prague, Czech Republic, 22–27 May 2011; pp. 1009–1012.
12. Chen, J.; Richard, C.; Honeine, P. Nonlinear unmixing of hyperspectral data based on a linear-mixture/nonlinear-fluctuation model. *IEEE Trans. Signal Process.* **2012**, *61*, 480–492. [CrossRef]
13. Halimi, A.; Honeine, P.; Bioucas-Dias, J. Hyperspectral Unmixing in Presence of Endmember Variability, Nonlinearity or Mismodelling Effects. *IEEE Trans. Image Process.* **2016**, *25*, 4565–4579. [CrossRef] [PubMed]
14. Chen, Y.; Lin, Z.; Zhao, X.; Wang, G.; Gu, Y. Deep learning-based classification of hyperspectral data. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2014**, *7*, 2094–2107. [CrossRef]
15. Wang, Q.; Liu, S.; Chanussot, J.; Li, X. Scene classification with recurrent attention of VHR remote sensing images. *IEEE Trans. Geosci. Remote Sens.* **2018**, *57*, 1155–1167. [CrossRef]
16. Li, J.; Li, X.; Huang, B.; Zhao, L. Hopfield neural network approach for supervised nonlinear spectral unmixing. *IEEE Geosci. Remote Sens. Lett.* **2016**, *13*, 1002–1006. [CrossRef]
17. Mitraka, Z.; Del Frate, F.; Carbone, F. Spectral unmixing of urban Landsat imagery by means of neural networks. In Proceedings of the 2015 Joint Urban Remote Sensing Event (JURSE), Lausanne, Switzerland, 30 March–1 April 2015; pp. 1–4.
18. Khajehrayeni, F.; Ghassemian, H. Hyperspectral unmixing using deep convolutional autoencoders in a supervised scenario. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2020**, *13*, 567–576. [CrossRef]
19. Palsson, B.; Sigurdsson, J.; Sveinsson, J.R.; Ulfarsson, M.O. Hyperspectral unmixing using a neural network autoencoder. *IEEE Access* **2018**, *6*, 25646–25656. [CrossRef]
20. Elkholy, M.M.; Mostafa, M.; Ebied, H.M.; Tolba, M.F. Hyperspectral unmixing using deep convolutional autoencoder. *Int. J. Remote Sens.* **2020**, *41*, 4799–4819. [CrossRef]
21. Wang, M.; Zhao, M.; Chen, J.; Rahardja, S. Nonlinear unmixing of hyperspectral data via deep autoencoder networks. *IEEE Geosci. Remote Sens. Lett.* **2019**, *16*, 1467–1471. [CrossRef]
22. Zhao, M.; Wang, M.; Chen, J.; Rahardja, S. Hyperspectral unmixing via deep autoencoder networks for a generalized linear-mixture/nonlinear-fluctuation model. *arXiv* **2019**, arXiv:1904.13017.
23. Lawrence, S.; Giles, C.L.; Tsoi, A.C.; Back, A.D. Face recognition: A convolutional neural-network approach. *IEEE Trans. Neural Netw.* **1997**, *8*, 98–113. [CrossRef]
24. Zhao, M.; Wang, M.; Chen, J.; Rahardja, S. Hyperspectral Unmixing for Additive Nonlinear Models with a 3-D-CNN Autoencoder Network. *IEEE Trans. Geosci. Remote Sens.* **2021**, *60*, 5509415. [CrossRef]
25. Li, H.; Borsoi, R.A.; Imbiriba, T.; Closas, P.; Bermudez, J.C.; Erdoğmuş, D. Model-Based Deep Autoencoder Networks for Nonlinear Hyperspectral Unmixing. *IEEE Geosci. Remote Sens. Lett.* **2021**, *19*, 5506105. [CrossRef]
26. Jin, Q.; Ma, Y.; Fan, F.; Huang, J.; Mei, X.; Ma, J. Adversarial autoencoder network for hyperspectral unmixing. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *13*, 1–15. [CrossRef]
27. Zhang, X.; Xu, J.; Yang, J.; Chen, L.; Zhou, H.; Liu, X.; Li, H.; Lin, T.; Ying, Y. Understanding the learning mechanism of convolutional neural networks in spectral analysis. *Anal. Chim. Acta* **2020**, *1119*, 41–51. [CrossRef]
28. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
29. Zhao, B.; Lu, H.; Chen, S.; Liu, J.; Wu, D. Convolutional neural networks for time series classification. *J. Syst. Eng. Electron.* **2017**, *28*, 162–169. [CrossRef]
30. Fawaz, H.I.; Lucas, B.; Forestier, G.; Pelletier, C.; Schmidt, D.F.; Weber, J.; Webb, G.I.; Idoumghar, L.; Muller, P.A.; Petitjean, F. Inceptiontime: Finding alexnet for time series classification. *Data Min. Knowl. Discov.* **2020**, *34*, 1936–1962. [CrossRef]
31. Peng, C.; Zhang, X.; Yu, G.; Luo, G.; Sun, J. Large kernel matters–improve semantic segmentation by global convolutional network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4353–4361.
32. Hao, S.; Wang, W.; Ye, Y.; Nie, T.; Bruzzone, L. Two-stream deep architecture for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **2017**, *56*, 2349–2361. [CrossRef]
33. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
34. Ammanouil, R.; Ferrari, A.; Richard, C.; Mathieu, S. Nonlinear unmixing of hyperspectral data with vector-valued kernel functions. *IEEE Trans. Image Process.* **2016**, *26*, 340–354. [CrossRef]
35. Févotte, C.; Dobigeon, N. Nonlinear hyperspectral unmixing with robust nonnegative matrix factorization. *IEEE Trans. Image Process.* **2015**, *24*, 4810–4819. [CrossRef] [PubMed]

36. Gao, L.; Han, Z.; Hong, D.; Zhang, B.; Chanussot, J. CyCU-Net: Cycle-consistency unmixing network by learning cascaded autoencoders. *IEEE Trans. Geosci. Remote Sens.* **2021**, *60*, 1–14. [CrossRef]
37. Swayze, G.; Clark, R.; King, T.; Gallagher, A.; Calvin, W. The US Geological Survey, Digital Spectral Library: Version 1: 0.2 to 3.0 mum. *Bull. Am. Astron. Soc.* **1993**, *25*, 1033.