IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 2, 2018, accepted March 10, 2018, date of publication March 22, 2018, date of current version May 24, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2818280

Hyperspectral Unmixing Using a Neural Network Autoencoder

BURKNI PALSSON, (Member, IEEE), JAKOB SIGURDSSON, (Member, IEEE), JOHANNES R. SVEINSSON, (Senior Member, IEEE), AND MAGNUS O. ULFARSSON, (Member, IEEE)

Faculty of Electrical and Computer Engineering, University of Iceland, 107 Reykjavik, Iceland

Corresponding author: Johannes R. Sveinsson (sveinsso@hi.is)

This work was supported in part by the Icelandic Research Fund under Grant 174075-05 and in part by the Postdoctoral Research Fund at the University of Iceland.

ABSTRACT In this paper, we present a deep learning based method for blind hyperspectral unmixing in the form of a neural network autoencoder. We show that the linear mixture model implicitly puts certain architectural constraints on the network, and it effectively performs blind hyperspectral unmixing. Several different architectural configurations of both shallow and deep encoders are evaluated. Also, deep encoders are tested using different activation functions. Furthermore, we investigate the performance of the method using three different objective functions. The proposed method is compared to other benchmark methods using real data and previously established ground truths of several common data sets. Experiments show that the proposed method compares favorably to other commonly used hyperspectral unmixing methods and exhibits robustness to noise. This is especially true when using spectral angle distance as the network's objective function. Finally, results indicate that a deeper and a more sophisticated encoder does not necessarily give better results.

INDEX TERMS Hyperspectral unmixing, autoencoder, deep learning, neural network, spectral angle distance, endmember extraction.

I. INTRODUCTION

Hyperspectral imaging is a rapidly growing field of remote sensing that has contributed significantly to Earth observations. In hyperspectral imaging, a special kind of sensor simultaneously acquires a large number of spatially coregistered images in many narrow, contiguous spectral bands, that are measured in calibrated radiance units which are subsequently converted into reflectance. Due to the high spectral resolution, it is possible to identify the material or materials present in each pixel of a hyperspectral image (HSI). This makes HSIs very useful in applications such as land cover classification, food crop monitoring, and geological exploration.

The high spectral resolution of HSIs is achieved at the cost of reduced spatial resolution. This is a direct consequence of the physical limitations of the sensor as the cumulative energy of incident light for each narrow band is too low to allow for many pixels. As a result of the low spatial resolution, each pixel of an HSI is usually a mixture of the reflectance spectra of more than one distinct material in the scene. The reflectance spectrum of a distinct or representative material in a scene is known as an endmember spectrum, and the relative proportion of an endmember in a pixel is called the abundance fraction of that endmember.

The task of hyperspectral unmixing (HSU) [1] is to simultaneously estimate the endmembers in the HSI and also their fractional abundances, i.e., the abundance vectors, for every pixel in the HSI. Thus, the spectra of pixels are decomposed into a weighted combination of the endmembers. Unmixing methods depend on the underlying mixing models, which are models of how the actual mixing of the spectra takes place. There are two types of mixing models: non-linear and linear models. Non-linear models attempt to model the physical interactions between the light scattered by multiple materials in the scene where the interactions can be at a classical (multilayered) level, or at a microscopic (intimate) level [2], [3]. In contrast, linear models make the assumption that mixing takes place on a macroscopic scale and that the incident light only interacts with a single material. The mixing takes place within the instrument itself due to low spatial resolution [4].

Thus, assuming the linear mixing model (LMM), if there is more than one material within the boundaries of a pixel, the observed spectral vector of the pixel will be a weighted sum of the same number of endmembers representing the materials plus noise that comes from the instruments. The majority of current unmixing methods use the LMM.

In the last couple of decades, numerous unmixing methods have been introduced, and they can be classified into geometrical, statistical, and sparse regression methods. The geometric methods can further be categorized as pure pixel methods or minimum volume methods [4]–[8]. Good examples of methods representing these two categories are Vertex Component Analysis (VCA) [9], and Minimum Volume Simplex Analysis (MVSA) [10]. Statistical methods are another large category of unmixing methods where unmixing is most often formulated as a statistical inference problem [11]–[13]. Sparse regression based unmixing methods are based on the assumption that the observed spectra can be expressed as linear combinations of known spectral signatures which can be identified in spectral libraries, which makes them semi-supervised in nature [14]–[17]. Methods based on compressed sensing also belong to this category of methods [4], [18]–[20].

Following the success of deep learning methods in computer vision, image classification, and natural language processing, the last few years have seen a great increase in the application of deep learning methods for remote sensing data, e.g., [21], [22]. A review of the literature shows that deep learning has not been applied to the problem of HSU to the same extent as to other applications in remote sensing, such as classification. Recent applications of neural network methods for unmixing are [23], [24], and [25] where an autoencoder is used for abundance estimation, i.e., mapping input spectra to abundance fractions but not extracting the actual endmembers. In [26], a shallow symmetric, i.e., the encoder and decoder have tied weights, nonnegative sparse autoencoder is used to extract endmembers. The novelty of this method lies in the use of an automatic sampler with a local outlier factor and affinity propagation for intelligently selecting samples for the training set. Another very recent unmixing method using an autoencoder is [27], where a part-based autoencoder integrating a denoising part performs denoising on the input data and removes redundant hidden nodes automatically and thus estimates the number of endmembers in the HSI. Another work using an autoencoder can be found in [28], where an autoencoder cascade concatenates a marginalized denoising autoencoder and a nonnegative sparse autoencoder to solve the unmixing problem.

In this paper, we apply deep learning to the problem of HSU in the form of an autoencoder having a deep encoder, and where both the abundances sum to one constraint (ASC) and the nonnegative abundances constraint (ANC) are enforced using a custom layer and weight constraints. The method additionally exploits sparsity in the abundance vectors by employing a form of adaptive thresholding that is optimized w.r.t. the objective function of the network. This work is an extension of the work done in [29], which investigated the performance of a simpler autoencoder architecture and did not use the spectral angle distance (SAD) objective function. One important difference between the proposed method and traditional methods is that there are no tuning parameters other than the number of endmember spectra to estimate, also the implementation of the framework for the proposed method makes it very easy to use a custom objective function of arbitrary complexity. The method does not estimate the number of endmembers in an HSI. Hence the number of endmembers has to be given.

Almost all other deep learning based methods for HSU do not perform blind unmixing, i.e., estimate both the endmember spectra and their abundances. To the authors' best knowledge, the only deep learning methods that perform blind unmixing, are the methods in [26]–[29]. The proposed method differs mainly from these methods by having both a deep encoder and being able to exploit the sparsity of abundances through a layer using a custom activation function instead of explicit sparsity regularization. As a result, it does not have a symmetrical architecture using tied weights for the encoder and decoder. Additionally, we test many different activation functions for the encoding part of the autoencoder and use three different objective functions, whereas most of the works using autoencoders for unmixing use the mean squared error (MSE) objective function.

The performance of the autoencoder is evaluated on real data sets using three different objective functions: MSE, spectral information divergence (SID), and SAD. Also, we investigate how different activation functions in the encoder affect its performance. Experiments show that both the SID and SAD objective functions give significantly better results than the MSE objective function, and having a deep encoder gives slightly better results than architectures with a shallow encoder.

The outline of the paper is as follows. In Section 2, the method will be described in detail, in Section 3 we will present the results of the experiments, and finally, in Section 4, conclusions will be drawn.

A. NOTATION

In the paper, the following notation will be used. $\boldsymbol{X} \in \mathbb{R}^{B\times P}$ denotes the input HSI having P pixels and B spectral bands, and $\boldsymbol{x}_p$ will be used for the spectra of pixel number p where $p = 1, \ldots, P$. The number of endmembers will be denoted by R. The endmembers themselves will be denoted by $\boldsymbol{m}_r$, where $r = 1, \ldots, R$, and the matrix having the endmembers as columns will be denoted $\boldsymbol{M} \in \mathbb{R}^{B\times R}$. The abundance vector for pixel p will be denoted by $\boldsymbol{s}_p = [s_{1,p}, \ldots, s_{R,p}]^T$, and the matrix having all abundance vectors as columns will be denoted using $\boldsymbol{S} \in \mathbb{R}^{R\times P}$. The activations of layer l in a neural network will be denoted by $\boldsymbol{a}^{(l)}$, where $l = 0, \ldots, L$ and L is the total number of layers (the input layer is not counted as a layer and is thus layer number 0). The activation of unit i in layer l will be denoted by $a_i^{(l)}$. The weights of layer l in the network will be denoted by $\boldsymbol{W}^{(l)}$.

II. PROBLEM FORMULATION

We are given P observed spectra, each having B bands. After estimating the number of endmembers in the scene as R, we assume the LMM

$$\boldsymbol{x}_p = \sum_{r=1}^{R} s_{r,p}\boldsymbol{m}_r + \boldsymbol{n}_p = \boldsymbol{M}\boldsymbol{s}_p + \boldsymbol{n}_p, \tag{1}$$

where $\boldsymbol{n}_p$ is noise and we assume that

$$s_{r,p} \geq 0 \qquad \forall r, \quad \forall p \text{ (ANC)}, \tag{2a}$$

$$\sum_{r=1}^{R} s_{r,p} = 1 \quad \forall p \text{ (ASC)}. \tag{2b}$$

If we consider the whole image at once, (1) can be written as

$$\boldsymbol{X} = \boldsymbol{M}\boldsymbol{S} + \boldsymbol{N}. \tag{3}$$

The problem investigated in this paper is the estimation of the endmember matrix $\boldsymbol{M}$ and the abundance matrix $\boldsymbol{S}$ in (3), given an HSI $\boldsymbol{X}$, by interpreting the problem as blind unmixing, and solving it in an unsupervised manner using an autoencoder. We will show that the LMM constrains the possible architecture of the autoencoder.

A. THE PROPOSED METHOD

An autoencoder is a feedforward neural network that is trained to reproduce its input by learning the identity function. The autoencoder can be considered to consist of two parts: An encoder, $\mathcal{G}_E\colon \mathbb{R}^{B\times 1} \to \mathbb{R}^{R\times 1}$, which encodes the input $\boldsymbol{x}_p$ to a hidden representation, $\mathcal{G}(\boldsymbol{x}_p) = \boldsymbol{h}_p \in \mathbb{R}^{R\times 1}$, and a decoder, $\mathcal{G}_D\colon \mathbb{R}^{R\times 1} \to \mathbb{R}^{B\times 1}$, which decodes the hidden representation $\boldsymbol{h}_p$ to an approximation of the input, $\mathcal{G}_D(\boldsymbol{h}_p) = \hat{\boldsymbol{x}}_p$. The network is thus trained using backpropagation to minimize the loss function,

$$\mathcal{L}(\boldsymbol{x}_p, \mathcal{G}_D(\mathcal{G}_E(\boldsymbol{x}_p))), \tag{4}$$

where $\mathcal{L}(\cdot,\cdot)$ is some measure of the discrepancy between the input and the output. By having a hidden layer with only a few units as the final layer of the encoder, it is mapping spectral vectors of length B into a latent space. The decoder then uses the latent vectors to reconstruct the input spectrum as faithfully as possible. Autoencoders where the dimensionality of the hidden representation (the code) is lower than the input/output dimension are known as under-complete autoencoders [30]. Fig. 1 shows a high level view of the proposed autoencoder.

Since the output $\hat{\boldsymbol{x}}_p$ of the network is the reconstruction of the input, $\boldsymbol{x}_p$, the output layer of the network (the last layer of the decoder, $\mathcal{G}_E$) has to have a linear activation function (a rectified linear unit [31] (ReLU) would also suffice). The action of this layer is thus the following linear transformation,

$$\hat{\boldsymbol{x}}_p = \boldsymbol{W}^{(L)}\boldsymbol{a}^{(L-1)}, \tag{5}$$

where $\boldsymbol{a}^{(L-1)}$ are the activations of the previous layer, and $\boldsymbol{W}^{(L)}$ are the weights of the output layer. If we compare (5) to the LMM (1), it is evident that the activations vector $\boldsymbol{a}^{(L-1)}$ has to be of dimension $R \times 1$, and $\boldsymbol{W}^{(L-1)}$ has to be a $B \times R$ matrix, where B is the number of bands. This leaves us with no flexibility regarding the structure of the decoder. It has to be a linear transformation $\mathcal{G}_D : \mathbb{R}^{R\times 1} \to \mathbb{R}^{B\times 1}$. This means that we have to interpret the hidden representation $\boldsymbol{h}_p = \mathcal{G}_E(\boldsymbol{x}_p) \in \mathbb{R}^{R\times 1}$ as the abundance vector for the input spectrum $\boldsymbol{x}_p$, and the weights of the decoder, $\boldsymbol{W}_D = \boldsymbol{W}^{(L)} \in \mathbb{R}^{B\times R}$, as a matrix whose columns are the endmembers. We have, however, much more flexibility regarding the encoder part of the network. The only requirements the LMM model places on the encoder is that its last layer must have R units. It can have many layers with non-linear activations, in contrast to the decoder which must be a single layer with all units having linear activation.

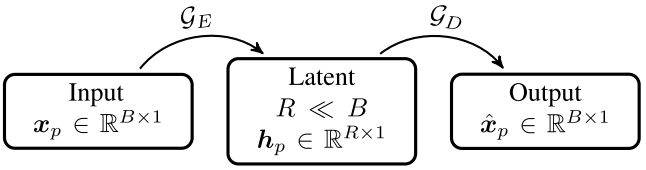

FIGURE 1. A high-level view of the proposed autoencoder. The latent space is the space of the representations of the input signal after the encoder has performed nonlinear dimensionality reduction on the input. Its dimensionality is equal to the number of endmembers.

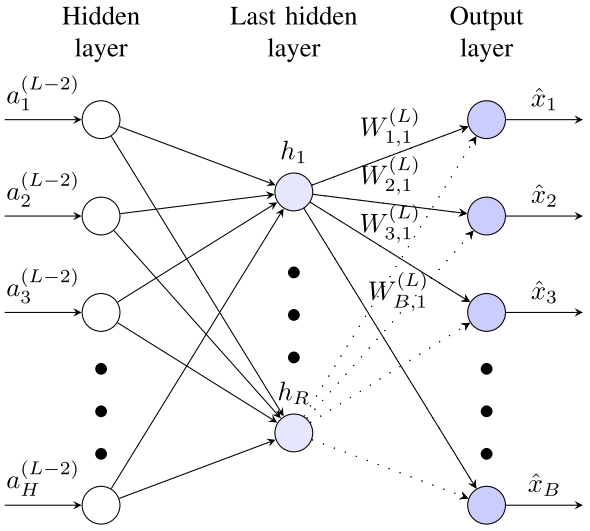

FIGURE 2. A schematic of the autoencoder showing the last two layers of the encoder, and the decoder (output layer with linear activations). At the end of training, the activations of the last hidden layer are the abundance fractions, and the network's weights connecting the last hidden layer and the output layer are the endmembers.

After training, the autoencoder has effectively performed blind unmixing on the input image $\boldsymbol{X}$ according to (3), and we can extract the matrix $\boldsymbol{S}$ as the matrix of the hidden activations $\boldsymbol{h}_i$ for all the P pixels, and the endmember matrix $\boldsymbol{M}$ as the decoder's weights. Fig. 2 shows a simplified schematic of the autoencoder. The figure shows the last two layers of the encoder, and the decoder which is simply the output layer. The activations $\boldsymbol{h} = [h_1, \ldots, h_R]^T$ of the units in the last hidden layer are the abundance fractions. A more detailed schematic of the proposed method is shown in Fig. 3.

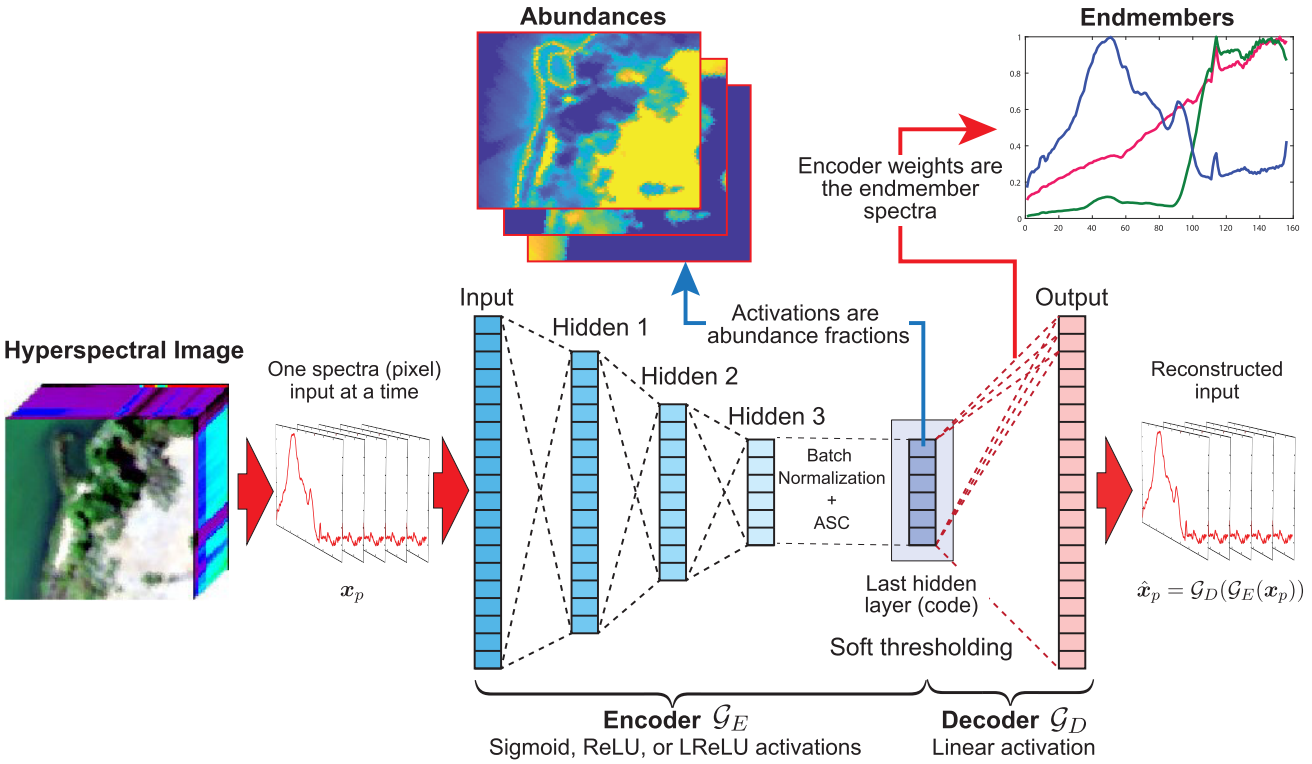

FIGURE 3. A graphical representation of the proposed method. The autoencoder is trained on all the spectra in the HSI for a number of epochs. After training, abundance maps can be extracted as the activations of the last hidden layer for each input spectra, and the weights of the decoder are the endmember spectra.

The simple structure of the decoder affects the performance of the autoencoder in reconstructing the input. Not being able to match a sophisticated deep encoder with a matching deep decoder limits the advantages of using a complex encoder instead of a more simple one. However, our experiments still show that a deep encoder can often be better than a simple one.

B. THE ENCODER

In this work, an autoencoder having both a deep and a shallow encoder was used and results obtained using both configurations were compared. In the following, we will use g to denote the activation function of hidden layers that are not utility layers. Utility layers are layers which have the same number of units as the previous layer and where each unit performs some transformation on the output of a corresponding unit in the previous layer, i.e., they are not fully connected. Table 1 lists the layers of the encoder. A more detailed description of the layers will be given below.

Hidden layers 1 to 4 all have the same activation function g, where g can be the Sigmoid [32], ReLU, or the Leaky ReLU (LReLU) [33] activation function. They apply the transformation

$$\boldsymbol{a}^{(l)} = g(\boldsymbol{W}^{(l)}\boldsymbol{a}^{(l-1)}), \tag{6}$$

to their input $\boldsymbol{a}^{(l-1)}$ from the previous layer, so together they reduce the dimensionality of the input $\boldsymbol{a}^{(0)} = \boldsymbol{x}_p$ in steps from B to R. It is essential that the activation function g is not linear, because then the encoder would simply be performing principal components analysis (PCA) dimensionality reduction. In our experiments, g was the Sigmoid, ReLU, or LReLU. The LReLU function is considered to be a better activation function than the standard ReLU function because it has nonzero gradient for all inputs [33].

TABLE 1. The layers of the encoder. The activation function g can be one of several types. Utility activation means that the layer performs some transform other than normal activation on the outputs of the previous layer.

Layer #	Layer Type	Activation function	# Units
1	Input	Linear	B
2	Hidden	g	9R
3	Hidden	g	6R
4	Hidden	g	3R
5	Hidden	g	R
6	Batch Normalization	Utility	R
7	Dynamical Soft Thresholding	RelU or LReLU	R
8	Enforces ASC	Utility	R
9	Gaussian Dropout	Utility	R

Hidden layer 5 is a Batch Normalization layer. Batch Normalization normalizes the activations of the previous layer at each batch, i.e., it applies a transformation that maintains the mean activation of every unit close to zero, and then applies scaling and shifting, parametrized by two learnable parameters, γ and β. If we let $\boldsymbol{a}_i^{(5)}, i = 1, \ldots, m$ be the values of the inputs from the previous layer for a batch $\mathcal{B}$, we can

write the effect of the Batch Normalization layer as

$$\boldsymbol{a}_i^{(6)} = \mathrm{BN}_{\gamma,\beta}(\boldsymbol{a}_i^{(5)}) = \gamma \hat{\boldsymbol{a}}_i^{(5)} + \beta, \tag{7}$$

where

$$\hat{\boldsymbol{a}}_i^{(5)} = \frac{\boldsymbol{a}_i^{(5)} - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}},$$
$$\mu_{\mathcal{B}} = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{a}_i^{(5)},$$
$$\sigma_{\mathcal{B}}^2 = \frac{1}{m}\sum_{i=1}^{m} (\boldsymbol{a}_i^{(5)} - \mu_{\mathcal{B}})^2,$$

γ and β are learnable parameters, and ϵ is a very small number. Batch Normalization whitens the data which is well known for speeding up learning in neural networks [34], and reduces so called internal covariance shift, which is a term used for the change of a layer's activations distribution over time [35].

Hidden layer 6 is a sparsity enhancing layer that seeks to exploit sparsity in abundance maps by using a soft thresholding ReLU activation function with a dynamic threshold α that is a learnable parameter for each unit in the layer. If we denote this activation function by $\theta_{\boldsymbol{\alpha}}$ then this layer is dynamically thresholding the abundance fractions, which are the input from the previous layer, according to

$$\boldsymbol{a}^{(7)} = \theta_{\boldsymbol{\alpha}}(\boldsymbol{a}^{(6)}) = \max(\mathbf{0}, \boldsymbol{a}^{(6)} - \boldsymbol{\alpha}), \tag{8}$$

where $\boldsymbol{\alpha}$ is an $R \times 1$ vector.

Layer 8 enforces the ASC. It performs the following transform on the inputs from the previous layer

$$a_i^{(8)} = \frac{a_i^{(7)}}{\sum_{j=1}^{R} a_j^{(7)}}, \tag{9}$$

which enforces the ASC. For convenience, we will denote the action of this layer using an operator notation, so (9) can be written as

$$\boldsymbol{a}^{(8)} = \mathbf{H}\boldsymbol{a}^{(7)}.$$

Layer 9, which is the last layer of the encoder, applies multiplicative Gaussian noise to its inputs. White noise injection into input and hidden layers of neural networks has been found to act as a powerful regularizer [36]. This type of layer is sometimes referred to as a Gaussian Dropout layer. Since it is a regularization layer, it is only active during training of the network.

The encoding of the input $\boldsymbol{x}_p = \boldsymbol{a}^{(0)}$ can now be written out fully by combining all the equations for all the layers as

$$\mathcal{G}_E(\boldsymbol{x}_p) = G(\mathbf{H}\theta_{\boldsymbol{\alpha}}(\mathrm{BN}(\boldsymbol{a}^{(5)}))), \tag{10}$$

where $\mathrm{BN}(\cdot)$ is Batch Normalization, $\boldsymbol{a}^{(5)}$ is given by

$$\boldsymbol{a}^{(5)} = g(\boldsymbol{W}^{(4)} g(\boldsymbol{W}^{(3)} g(\boldsymbol{W}^{(2)} g(\boldsymbol{W}^{(1)} \boldsymbol{a}^{(0)})))), \tag{11}$$

and the function G denotes the action of the Gaussian Dropout layer. This encoding of the input $\boldsymbol{x}_p$ is then extracted as the abundance fractions of the R endmembers. The autoencoder's reconstruction of its input after training can now be written as

$$\mathcal{G}_D(\mathcal{G}_E(\boldsymbol{x}_p)) = \boldsymbol{W}_D \mathbf{H}\theta_{\boldsymbol{\alpha}}(\mathrm{BN}(\boldsymbol{a}^{(5)})), \tag{12}$$

since the Gaussian Dropout layer is only active during training, and where $\boldsymbol{W}_D$ contains the endmembers in its columns. The quantity $\mathbf{H}\theta_{\boldsymbol{\alpha}}(\mathrm{BN}(\boldsymbol{a}^{(5)}))$ is an $R \times 1$ vector containing the abundance fractions for the input spectra.

C. OBJECTIVE FUNCTIONS

In this paper, we used the MSE, SID and SAD measures as objective functions for the neural network. The MSE objective function is given by

$$J_{\mathrm{MSE}} = \frac{1}{P}\sum_{p=1}^{P} \|\boldsymbol{x}_p - \hat{\boldsymbol{x}}_p\|_2^2. \tag{13}$$

The SAD objective function is given by [37]

$$J_{\mathrm{SAD}} = \frac{1}{P}\sum_{p=1}^{P} \arccos\left(\frac{\langle \boldsymbol{x}_p, \hat{\boldsymbol{x}}_p \rangle}{\|\boldsymbol{x}_p\|_2 \, \|\hat{\boldsymbol{x}}_p\|_2}\right), \tag{14}$$

and measures the angle between spectra in their signal space in radians. Lastly, the SID objective function is given by [38]

$$J_{\mathrm{SID}} = \frac{1}{P}\sum_{p=1}^{P}\sum_{n=1}^{B} p_n \log\left(\frac{p_n}{q_n}\right) + \sum_{n=1}^{B} q_n \log\left(\frac{q_n}{p_n}\right), \tag{15}$$

where

$$p_n = \frac{\boldsymbol{x}_{i,n}}{\sum_{k=1}^{M} \boldsymbol{x}_{i,k}}, \quad q_n = \frac{\hat{\boldsymbol{x}}_{i,n}}{\sum_{k=1}^{M} \hat{\boldsymbol{x}}_{i,k}},$$

are estimates of the probability mass functions of the target and estimated spectra, respectively.

Both the SID and SAD objective functions differ from the MSE function in one important aspect. They are scale invariant while the MSE function is not. MSE sensitivity to the scale of spectra can be seen as a drawback regarding its use as an objective function in unmixing applications since it will discriminate between the same endmember in a scene based on its absolute magnitude. The scale invariance of the SID and the SAD objective functions could potentially lead to an incorrect scale of estimated endmembers, but as long as the ASC on abundances is enforced in the neural network, the relative scale of the endmember spectra is not affected.

III. EXPERIMENT RESULTS

In this section, we evaluate the proposed autoencoder method using real hyperspectral data. Different objective functions and activation functions will be evaluated and compared. The results obtained using the proposed method for each data set will then be compared to the methods tabulated in Table 2.

Table 3 summarizes all the different configurations of the autoencoder that will be used in the experiment.

In the experiment, all the configurations were tested with the three different objective functions and each configuration

TABLE 2. The benchmark methods used in comparison with the proposed method.

Method	Description
VCA	Vertex Component Analysis [9], which is a classic geometric method that works well when scenes contain pure pixels.
NMF	Nonnegative Matrix Factorization [39], which is a benchmark statistical method.
ℓ_1-NMF	Nonnegative sparse coding [40], which is an example of a sparse regularized NMF method.
$\ell_{1/2}$-NMF	$\ell_{1/2}$ sparsity constrained NMF [41]. This method uses the $\ell_{1/2}$ penalty and enforces the ASC.
DCD	A HSU algorithm using ℓ_1 sparse regularization and dyadic cyclic descent optimization (DCD) [42].
R-CoNMF	Robust collaborative nonnegative matrix factorization (R-CoNMF), which is a recent method that, in addition to determining endmembers and their abundance maps, also estimates the number of endmembers in a scene [43].

TABLE 3. Different Autoencoder configurations and their names.

Configuration Name	Encoder Layers	Encoder Activation	Soft Thresholding Activation
Sigmoid	9, 6, 3	Sigmoid	SReLU
Sigmoid-SLReLU	9, 6, 3	Sigmoid	SLReLU
ReLU	9, 6, 3	ReLU	SReLU
ReLU-SLReLU	9, 6, 3	ReLU	SLReLU
LReLU	9, 6, 3	LReLU	SReLU
LReLU-SLReLU	9, 6, 3	LReLU	SLReLU
Shallow	1	NA	SReLU

was run 50 times. The objective here is to see if different activation functions of a deep encoder will have any significant effect, and to see if there is anything to be gained from having a deep encoder instead of a shallow one. Also, to investigate the effects of replacing the sparsity-inducing soft thresholding ReLU layer with a corresponding soft thresholding LReLU layer. The numbers in the second column of Table 3 indicate both the number of hidden layers in the encoder (the count of numbers) and their sizes (number of hidden units) as multiples of the encoding dimension (the number of endmembers).

Evaluating and comparing HSU methods using real hyperspectral data is currently challenging because of the limited number of real HSIs with ground truth for endmember spectral signatures available (labeled HSIs). A recent effort to remedy this, described in detail in [44], has lead to 15 common HSIs along with 18 versions of ground truth for them being made publicly available on the web[1] for use in HSU research. For many scenes, the endmember spectral signatures are chosen manually by the authors from existing spectra in the image based on their expertise and understanding. In some cases, there can be pure pixels representing certain endmembers. After the ground truth endmembers have been selected, their validity is verified by comparing them to other ground truths in the literature, by doing a consistency check with the abundance labeling, and by relying on the authors' expertise.

The data sets used in this paper are summarized in Table 4. They consist of small regions of interest in the case of the Samson and the Jasper Ridge HSIs, and the whole Urban HSI. All of these HSIs are popular and widely used data sets in HSU research. Some corrupted, and water-absorption bands have been removed in the case of the Jasper Ridge and the Urban data sets. More detailed description of these data sets can be found in [44]. In order to compare our proposed method to other state-of-the-art methods using these data sets, we use the above-mentioned ground truths and calculate the SAD between the endmember spectra obtained and the given ground truth endmember spectra.

A. AUTOENCODER ARCHITECTURE

There are a number of hyperparameters in the proposed method that need to be set. Most of them concern the architecture of the network itself, i.e., the number of hidden layers and their sizes. Others are parameters for the gradient descent optimizer used and regularization parameters such as the Gaussian Dropout rate. The hyperparameters are the following: the number of hidden layers of the encoder, the number of units in each layer, the batch size, the optimizer used for gradient descent (and its parameters), and the rate of the Gaussian Dropout. The only real hyperparameter from the HSU viewpoint that needs to be estimated is R, the number of endmembers.

The current values for the number of layers and number of units of each hidden layer of the encoder were roughly determined by systematic experimentation. A rather low rate of Gaussian Dropout was selected based on experimentation. The choice of optimizer was found to have somewhat more effect than was expected. A number of common optimizers for gradient descent such as Rmsprop [45], Adam [46], Adadelta [47], Adamax [47], were tested, and the Adam optimizer was found to give the best results.

The single hyperparameter that had the greatest effect on the results of the autoencoder was the batch size. The batch size is the number of random training samples utilized in a single iteration for gradient descent. Experimentation showed that the batch size needed to be very low or around 5 for all real data sets tested except for Samson where the batch size was set to 20. Too large or too small batch size usually resulted in a poor solution for the least represented endmember, i.e., the endmember present in the fewest pixels of the HSI. A good example of this behavior is the ''road'' endmember in the Jasper Ridge HSI. The batch size is effectively the sample size used to estimate the gradient for backpropagation, hence a small batch size results in a noisy gradient. A noisy gradient could help the network avoid ending in saddle points or local minima. This problem of underrepresented endmembers needs further investigation.

[1] Available here: http://lesun.weebly.com/hyperspectral-data-set.html

TABLE 4. The data sets used.

Data sets	Image size # row	# column	Number of bands # all bands	# selected bands	# Endmembers	Top left pixel in original image
Samson	95	95	156	156	3	(232,93)
Jasper Ridge	100	100	224	198	4	(1, 272)
Urban	307	307	224	162	6	(1,1)

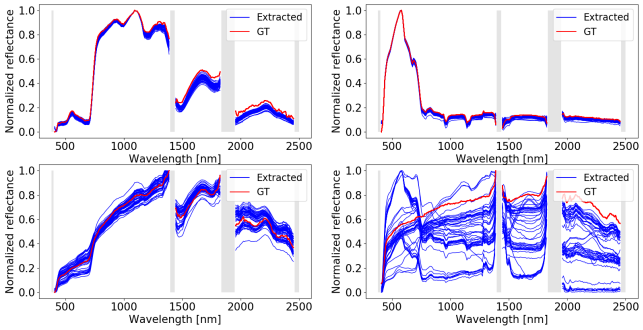

FIGURE 4. Variability of extracted endmembers for the Jasper Ridge data set as extracted by the autoencoder using the Sigmoid configuration and the SAD objective function. The red curves are the ground truth used, and the blue curves are the extracted endmembers from 50 runs.

The variability of all four endmembers estimated for the Jasper Ridge data set as extracted using the Sigmoid configuration of the autoencoder is shown in Fig. 4. The figure shows clearly that the autoencoder has some difficulty with one endmember, which is the underrepresented ''road'' endmember. The number of samples of this endmember during training is much lower than for the other endmembers, making the endmember not as well defined and stable as the others. The underrepresented endmember often appears to be a mixture of at least two other endmembers.

B. PERFORMANCE OF DIFFERENT AUTOENCODER CONFIGURATIONS BY DATA SETS

In this subsection, we will evaluate the performance of the autoencoder for different configurations and objective functions. The results for Samson, Jasper Ridge, and Urban data sets are shown in Fig. 5, Fig. 6, and Fig. 7, respectively.

The three figures show clearly that the SAD objective function gives the best results, followed closely by the SID objective function. The MSE objective function performs relatively poorly compared to the two other in all cases except for the Urban data set with a shallow encoder configuration. In almost all cases, the results for the SAD objective function have less variance than the results for the SID objective function.

The difference between different configurations is much less clear than the difference between different objective functions. Based on these three data sets, it is not possible to draw any conclusions about which configuration gives the best results as this seems to be data set dependent. The best results are obtained using a deep encoder, but it is only for the Urban data set that the difference between the deep encoder and the shallow encoder seems to be significant. Another observation is that the ReLU and LReLU activation functions

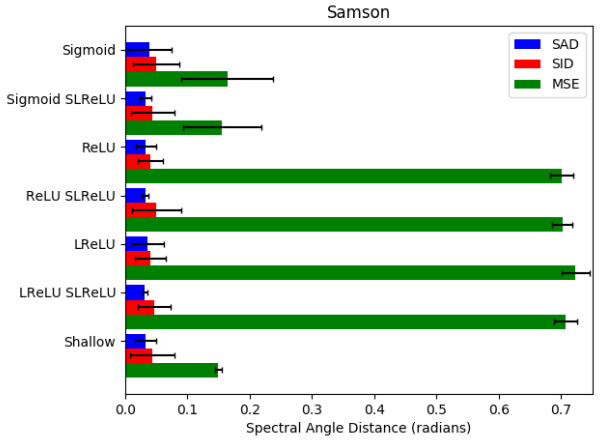

FIGURE 5. Performance of different autoencoder configurations for the Samson data set measured as SAD from the ground truth.

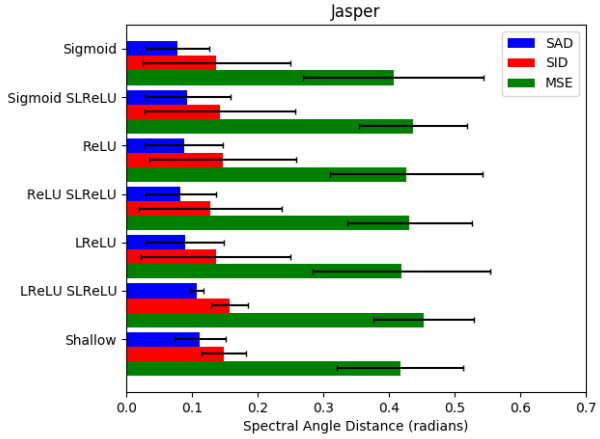

FIGURE 6. Performance of different autoencoder configurations for the Jasper Ridge data set measured as SAD from the ground truth.

seem to make the autoencoder more stable and consistent as the results for these configurations have less variance than the ones using the Sigmoid activation function. Fig. 5 for the Samson data set does, however, show clearly that for this particular data set and batch size, the MSE objective function gives markedly worse results when the activation function is not the Sigmoid function.

Perhaps the most important conclusion that can be drawn from these results is the fact that a deeper and more complex encoder does not give any significant advantage over a shallow one. This should not really be surprising since the decoder is only a single layer and is not able to take advantage of a powerful encoder.

TABLE 5. Mean Spectral Angle Distance (SAD) in radians with standard deviation from ground truth.

	Methods						
Data Set	Autoencoder	VCA	NMF	ℓ_1-NMF	$\ell_{1/2}$-NMF	CoNMF	DCD
Samson	**0.031±0.004**	0.10±0.08	0.14±0.13	0.11±0.06	0.09±0.09	0.1705±0.0004	0.0472±0.0001
Jasper	**0.078±0.05**	0.36±0.05	0.33±0.03	0.30±0.04	0.18±0.08	0.150±0.004	0.1413±0.0007
Urban	**0.151±0.009**	0.54±0.07	0.32±0.04	0.30±0.03	0.16±0.03	0.486±0.004	0.45±0.04

TABLE 6. Robustness to noise. Mean SAD in radians from ground truth with standard deviation for data set Jasper Ridge.

SNR	Autoencoder	VCA	NMF	ℓ_1-NMF	$\ell_{1/2}$-NMF	CoNMF	DCD
10	0.10±0.05	0.25±0.03	0.311±0.006	0.287±0.006	0.340±0.003	0.300±0.004	0.1532±0.0000
20	0.09±0.05	0.18±0.02	0.21±0.01	0.225±0.009	0.232±0.003	0.2046±0.0000	0.1084±0.0003
30	0.10±0.06	0.35±0.04	0.30±0.05	0.29±0.05	0.25±0.08	0.153±0.008	0.1151±0.0006
40	0.10±0.05	0.36±0.05	0.31±0.04	0.25±0.03	0.20±0.07	0.149±0.002	0.131±0.002

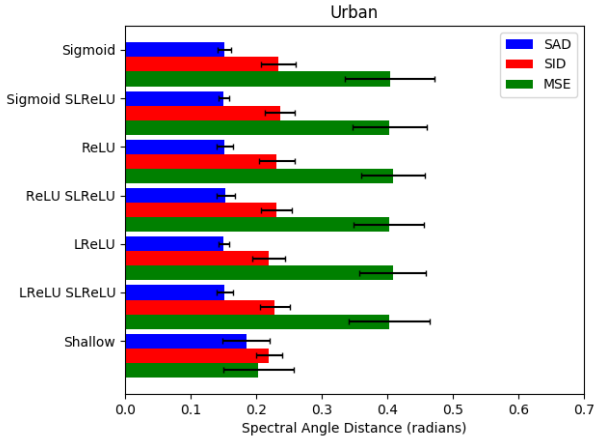

FIGURE 7. Performance of different autoencoder configurations for the Urban data set measured as spectral angle distance from the ground truth.

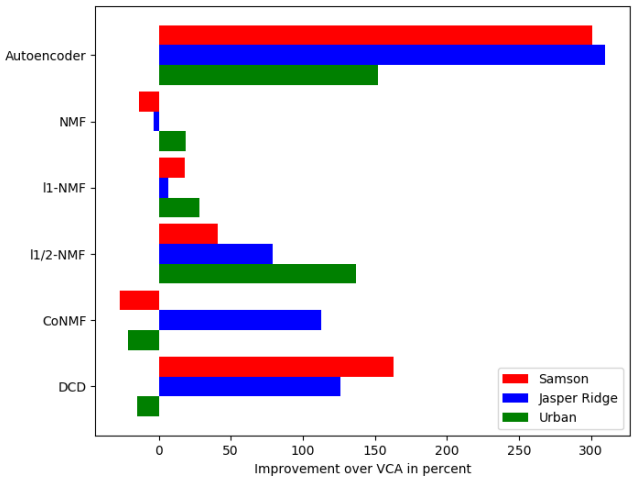

FIGURE 8. Performance of NMF, ℓ_1-NMF, $\ell_{1/2}$-NMF, CoNMF and DCD relative to method VCA.

C. COMPARISON WITH OTHER METHODS

In this subsection the performance of the proposed method will be compared to the methods NMF, ℓ_1-NMF, $\ell_{1/2}$-NMF, CoNMF, and DCD described at the beginning of this section.

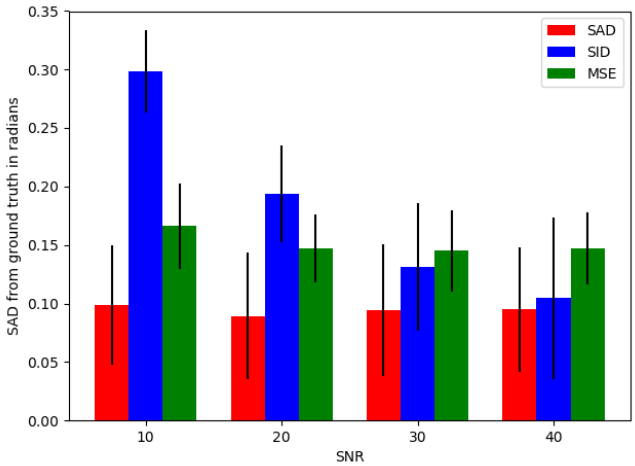

FIGURE 9. Mean SAD for all three autoencoder objective functions as function of SNR.

Table 5 shows the average SAD of all the endmembers of each data set from the ground truth established in [44]. All the values in the table are the mean of 50 runs, and the values in bold typeface are the lowest values for each data set. VCA was used to initialize the endmember matrices and fully constrained least squares regression was used to initialize the abundance matrices for all NMF methods, and also the DCD method. The proposed method performs substantially better than all the comparison methods for all the data sets tested. Fig. 8 is a graphical representation of the performance of each method relative to the VCA method. The bars indicate percentage improvement over VCA.

D. ROBUSTNESS TO NOISE

Here we investigate the robustness of the proposed method to noise and compare it to the other benchmark methods. For this experiment we used the Jasper Ridge data set and added low pass filtered Gaussian noise, resulting in a signal to noise ratio (SNR) of 10, 20, 30, and 40 dB, respectively. Only the Sigmoid configuration of the autoencoder was used. The results are shown in Table 6. The figures in the table are mean SAD of 50 runs in radians from the ground truth used and the standard deviations. A bar chart of the robustness to noise results for the autoencoder for

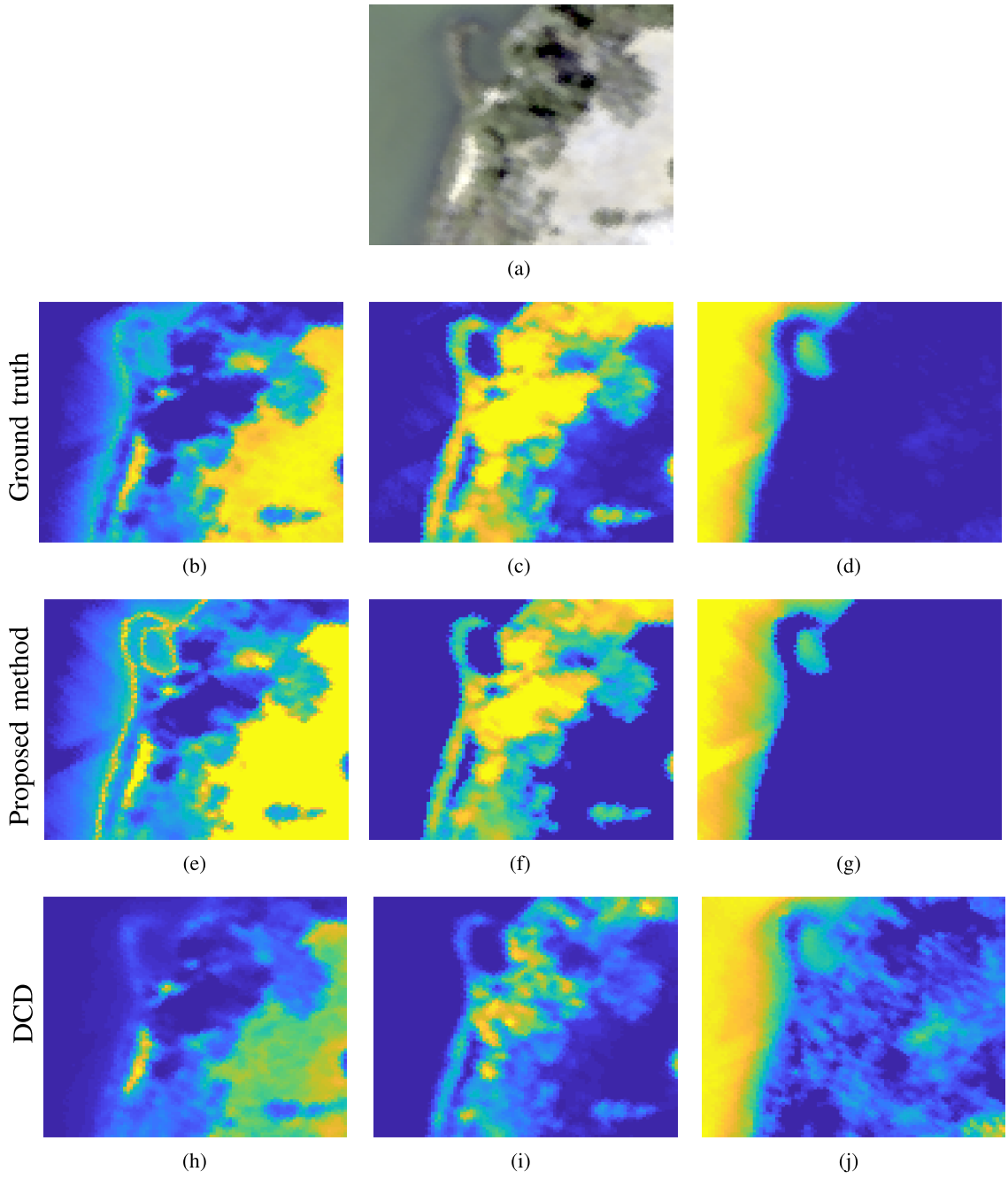

FIGURE 10. Estimated abundance maps for the Samson data set. The topmost image is a simulated RGB image of the data set. The images in the second row are the ground truth abundance maps. The images in the third row were obtained by the proposed method. The images in the bottom row were obtained by the DCD method. The images in the first column represent the "Soil" endmember, the images in the second column the "Trees" endmember, and the third column the "Water" endmember. The text in the subfigure captions gives the sparsity of the abundance maps as the percentage of pixels with abundance fractions less than 0.0001. (a) Simulated RGB image. (b) Sparsity(%) 15.19. (c) Sparsity(%) 18.58. (d) Sparsity(%) 46.01. (e) Sparsity(%) 8.81. (f) Sparsity(%) 50.24. (g) Sparsity(%) 71.07. (h) Sparsity(%) 7.30. (i) Sparsity(%) 20.86. (j) Sparsity(%) 8.76.

the three objective functions used can be seen in Fig. 9. Of these three, SAD clearly exhibits the most robustness to noise and gives similar results for all SNR values used. The SID objective function seem to have much less tolerance to noise than both the SAD and the MSE objective functions.

E. ABUNDANCE MAPS

The metric we have used to compare the different methods has been the mean SAD from a previously established ground truth for endmember spectral signatures. We have not evaluated the abundance maps quantitatively in this paper. We conclude this section by evaluating the estimated

abundance maps of the proposed method for the Samson data set. We compare it qualitatively to the abundance map ground truth for the Samson data set and the abundance maps estimated by the best of the other methods as determined by visual inspection. Of all the methods, the $\ell_{1/2}$ method was found to give the most sparse abundance maps. However, they were not found to be the best by visual inspection.

The comparison method obtaining the best abundance maps for the Samson data set by visual inspection was the DCD method. It was also the method that achieved the lowest mean SAD from the ground truth endmembers of all the comparison methods. Fig. 10 shows the estimated abundance maps for the Samson data set along with a simulated RGB image. The figures in the second row labeled (b), (c), and (d) are the ground truth abundance maps that correspond to the ground truth endmember spectral signatures used in this work. The figures in the third row labeled (e), (f) and (g) are (one solution) the estimated abundance maps produced by the proposed method. The configuration used is the Sigmoid configuration. As can be seen from the figure, these abundance maps are very visually similar to the ground truth maps, but they have substantially higher sparsity for the ''water'' and the ''trees'' endmembers. The last row of figures are the abundance maps estimated by the DCD method. They all have substantially lower sparsity than the other maps and are not as visually similar to the ground truth maps. The ''water'' abundance map exhibits considerable mixing and hence low sparsity.

IV. CONCLUSIONS

In this paper, we have proposed a deep learning based method for blind hyperspectral unmixing in the form of an asymmetrical autoencoder. Various configurations of the decoder part of the autoencoder were tested and compared. Three different objective functions were tested with all configurations. In this work, we focused on endmember spectra extraction and used previously established endmember ground truth for common data sets for evaluation. The comparison methods are some of the most benchmarked methods in HSU. Experimental results show clearly that the SAD objective function yields better results than the SID or the MSE objective function. The results also indicate that a deep and more complex encoder does not give markedly better results than a single layer encoder. Experiments with data with added noise showed very good robustness to noise, especially using the SAD objective function. Overall, the proposed method was found to perform substantially better than the comparison methods for all three data sets tested.

REFERENCES

[1] N. Keshava and J. F. Mustard, ''Spectral unmixing,'' *IEEE Signal Process. Mag.*, vol. 19, no. 1, pp. 44–57, Jan. 2002.

[2] R. Heylen, M. Parente, and P. Gader, ''A review of nonlinear hyperspectral unmixing methods,'' *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 6, pp. 1844–1868, Jun. 2014.

[3] N. Dobigeon, J.-Y. Tourneret, C. Richard, J. C. M. Bermudez, S. McLaughlin, and A. O. Hero, ''Nonlinear unmixing of hyperspectral images: Models and algorithms,'' *IEEE Signal Process. Mag.*, vol. 31, no. 1, pp. 82–94, Jan. 2014.

[4] J. M. Bioucas-Dias *et al.*, ''Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches,'' *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 5, no. 2, pp. 354–379, Apr. 2012.

[5] J. Sigurdsson, M. O. Ulfarsson, and J. R. Sveinsson, ''Total variation and ℓ_q based hyperspectral unmixing for feature extraction and classification,'' in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2015, pp. 437–440.

[6] J. Sigurdsson, M. O. Ulfarsson, and J. R. Sveinsson, ''Blind hyperspectral unmixing using total variation and ℓ_q sparse regularization,'' *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 11, pp. 6371–6384, Nov. 2016.

[7] L. Miao and H. Qi, ''Endmember extraction from highly mixed data using minimum volume constrained nonnegative matrix factorization,'' *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 3, pp. 765–777, Mar. 2007.

[8] S. Zhang, A. Agathos, and J. Li, ''Robust minimum volume simplex analysis for hyperspectral unmixing,'' *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 11, pp. 6431–6439, Nov. 2017.

[9] J. M. P. Nascimento and J. M. Bioucas-Dias, ''Vertex component analysis: A fast algorithm to unmix hyperspectral data,'' *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 4, pp. 898–910, Apr. 2005.

[10] J. Li, A. Agathos, D. Zaharie, J. M. Bioucas-Dias, A. Plaza, and X. Li, ''Minimum volume simplex analysis: A fast algorithm for linear hyperspectral unmixing,'' *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 9, pp. 5067–5082, Sep. 2015.

[11] J. M. P. Nascimento and J. M. Bioucas-Dias, ''Hyperspectral unmixing based on mixtures of Dirichlet components,'' *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 3, pp. 863–878, Mar. 2012.

[12] N. Dobigeon, S. Moussaoui, M. Coulon, J.-Y. Tourneret, and A. O. Hero, ''Joint Bayesian endmember extraction and linear unmixing for hyperspectral imagery,'' *IEEE Trans. Signal Process.*, vol. 57, no. 11, pp. 4355–4368, Nov. 2009.

[13] O. Eches, N. Dobigeon, C. Mailhes, and J.-Y. Tourneret, ''Bayesian estimation of linear mixtures using the normal compositional model. Application to hyperspectral imagery,'' *IEEE Trans. Image Process.*, vol. 19, no. 6, pp. 1403–1413, Jun. 2010.

[14] M.-D. Iordache, J. Bioucas-Dias, and A. Plaza, ''Sparse unmixing of hyperspectral data,'' *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 6, pp. 2014–2039, Jun. 2011.

[15] M.-D. Iordache, J. Bioucas-Dias, and A. Plaza, ''Total variation spatial regularization for sparse hyperspectral unmixing,'' *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 11, pp. 4484–4502, Nov. 2012.

[16] M.-D. Iordache, J. M. Bioucas-Dias, and A. Plaza, ''Collaborative sparse regression for hyperspectral unmixing,'' *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 1, pp. 341–354, Jan. 2014.

[17] M.-D. Iordache, J. Bioucas-Dias, A. Plaza, and B. Somers, ''MUSIC-CSR: Hyperspectral unmixing via multiple signal classification and collaborative sparse regression,'' *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 7, pp. 4364–4382, Jul. 2014.

[18] L. Zhang, W. Wei, Y. Zhang, F. Li, and H. Yan, ''Structured sparse BAYESIAN hyperspectral compressive sensing using spectral unmixing,'' in *Proc. 6th Workshop Hyperspectral Image Signal Process., Evol. Remote Sens. (WHISPERS)*, Jun. 2014, pp. 1–4.

[19] R. T. Albayrak, A. C. Gurbuz, and B. Gunyel, ''Compressed sensing based hyperspectral unmixing,'' in *Proc. 22nd Signal Process. Commun. Appl. Conf. (SIU)*, Apr. 2014, pp. 1438–1441.

[20] Xusu, ''Compressive sensing for endmember extraction,'' in *Proc. 2nd IEEE Int. Conf. Comput. Commun. (ICCC)*, Oct. 2016, pp. 1345–1348.

[21] Y. Chen, Z. Lin, X. Zhao, G. Wang, and Y. Gu, ''Deep learning-based classification of hyperspectral data,'' *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 6, pp. 2094–2107, Jun. 2014.

[22] F. Palsson, J. R. Sveinsson, and M. O. Ulfarsson, ''Multispectral and hyperspectral image fusion using a 3-D-convolutional neural network,'' *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 5, pp. 639–643, May 2017.

[23] J. Li, X. Li, B. Huang, and L. Zhao, ''Hopfield neural network approach for supervised nonlinear spectral unmixing,'' *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 7, pp. 1002–1006, Jul. 2016.

[24] Z. Mitraka, F. D. Frate, and F. Carbone, ''Spectral unmixing of urban landsat imagery by means of neural networks,'' in *Proc. Joint Urban Remote Sens. Event (JURSE)*, Mar. 2015, pp. 1–4.

[25] G. A. Licciardi and F. D. Frate, ''Pixel unmixing in hyperspectral data by means of neural networks,'' *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 11, pp. 4163–4172, Nov. 2011.

[26] Y. Su, A. Marinononi, J. Li, A. Plaza, and P. Gamba, "Nonnegative sparse autoencoder for robust endmember extraction from remotely sensed hyperspectral images," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2017, pp. 205–208.
[27] Y. Qu, R. Guo, and H. Qi, "Spectral unmixing through part-based nonnegative constraint denoising autoencoder," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2017, pp. 209–212.
[28] R. Guo, W. Wang, and H. Qi, "Hyperspectral image unmixing using autoencoder cascade," in *Proc. 7th Workshop Hyperspectral Image Signal Process., Evol. Remote Sens. (WHISPERS)*, Jun. 2015, pp. 1–4.
[29] F. Palsson, J. Sigurdsson, J. R. Sveinsson, and M. O. Ulfarsson, "Neural network hyperspectral unmixing with spectral information divergence objective," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2017, pp. 755–758.
[30] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org
[31] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," *J. Mach. Learn. Res.*, vol. 15, no. 4, pp. 315–323, 2011.
[32] J. Han and C. Moraga, *The Influence of the Sigmoid Function Parameters on the Speed of Backpropagation Learning*. Berlin, Germany: Springer, 1995, pp. 195–201.
[33] B. Xu, N. Wang, T. Chen, and M. Li. (May 2015). "Empirical Evaluation of Rectified Activations in Convolutional Network." [Online]. Available: https://arxiv.org/abs/1505.00853
[34] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, *Efficient BackProp*. Berlin, Germany: Springer, 2012, pp. 9–48.
[35] S. Ioffe and C. Szegedy. (Feb. 2015). "Batch normalization: Accelerating deep network training by reducing internal covariate shift." [Online]. Available: https://arxiv.org/abs/1502.03167
[36] Y. Li and F. Liu. (Dec. 2016). "Whiteout: Gaussian adaptive noise regularization in FeedForward neural networks." [Online]. Available: https://arxiv.org/abs/1612.01490
[37] S. Liang, *Comprehensive Remote Sensing*. Amsterdam, The Netherlands: Elsevier, 2017.
[38] C. Chang, "Spectral information divergence for hyperspectral image analysis," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, vol. 1. Jun. 1999, pp. 509–511.
[39] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, Oct. 1999.
[40] P. O. Hoyer, "Non-negative sparse coding," in *Proc. 12th IEEE Workshop Neural Netw. Signal Process.*, Jun. 2002, pp. 557–565.
[41] Y. Qian, S. Jia, J. Zhou, and A. Robles-Kelly, "Hyperspectral unmixing via $L_{1/2}$ sparsity-constrained nonnegative matrix factorization," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 11, pp. 4282–4297, Nov. 2011.
[42] J. Sigurdsson, M. O. Ulfarsson, J. R. Sveinsson, and J. M. Bioucas-Dias, "Sparse distributed multitemporal hyperspectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 11, pp. 6069–6084, Nov. 2017.
[43] J. Li, J. M. Bioucas-Dias, A. Plaza, and L. Liu, "Robust collaborative nonnegative matrix factorization for hyperspectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 10, pp. 6076–6090, Oct. 2016.
[44] F. Zhu. (Aug. 2017). "Hyperspectral unmixing: Ground truth labeling, datasets, benchmark performances and survey." [Online]. Available: https://arxiv.org/abs/1708.05125
[45] T. Tieleman and G. Hinton, "RmsProp: Divide the gradient by a running average of its recent magnitude," in *Proc. Neural Netw. Mach. Learn. (COURSERA)*, 2012, pp. 26–31.
[46] D. P. Kingma and J. Ba. (Dec. 2014). "Adam: A method for stochastic optimization." [Online]. Available: https://arxiv.org/abs/1412.6980
[47] M. D. Zeiler. (Dec. 2012). "ADADELTA: An adaptive learning rate method." [Online]. Available: https://arxiv.org/abs/1212.5701

BURKNI PALSSON (M'17) received the B.S. degree in mathematics, the B.S. degree in physics, and the B.S. degree in energy and environmental technology from the University of Iceland, Reykjavik, Iceland, in 1999 and 2012, respectively, where he is currently pursuing the M.S. degree in electrical engineering. His research interests include hyperspectral unmixing and classification in remote sensing and the development and applications of deep learning based methods in that field.

JAKOB SIGURDSSON (S'10–M'15) received the B.S. and M.S. degrees and the Ph.D. degree from the University of Iceland, in 2011 and 2015, respectively. He is currently a Post-Doctoral Researcher with the University of Iceland. His research interests include statistical signal- and image processing and remote sensing.

JOHANNES R. SVEINSSON (SM'02) received the B.S. degree from the University of Iceland, Reykjavík, and the M.S. and Ph.D. degrees from Queen's University, Kingston, ON, Canada, all in electrical engineering. He was with the Laboratory of Information Technology and Signal Processing from 1981 to 1982. He was a Visiting Research Student with the Imperial College of Science and Technology, London, U.K., from 1985 to 1986. He was with the Department of Electrical and Computer Engineering, Engineering Research Institute, as a Senior Member of research staff and a Lecturer, respectively, from 1991 to 1998. At Queen's University, he held teaching and research assistantships. He is currently a Professor with the Department of Electrical and Computer Engineering, University of Iceland. His current research interests are in systems and signal theory. He was a recipient of the Queen's Graduate Awards from Queen's University. He was a co-recipient of the 2013 IEEE GRSS Highest Impact Paper Award.

MAGNUS O. ULFARSSON (S'00–M'05) received the B.S. and the M.S. degrees from the University of Iceland in 2002, and the Ph.D. degree from the University of Michigan in 2007. He joined the University of Iceland in 2007, where he is currently a Professor. He has been affiliated with deCODE Genetics, Reykjavik, Iceland, since 2013. His research interests include statistical signal processing, genomics, medical imaging, and remote sensing.