UNIVERSITY OF ICELAND

Ph.D. Dissertation

in Electrical and Computer Engineering

Blind Hyperspectral Unmixing Using Autoencoders

Burkni Pálsson

September 2023

FACULTY OF ELECTRICAL AND COMPUTER ENGINEERING

Blind Hyperspectral Unmixing Using Autoencoders

Burkni Pálsson

Dissertation submitted in partial fulfillment of a *Philosophiae Doctor* degree in Electrical and Computer Engineering

Advisors
Jóhannes R. Sveinsson
Magnús Örn Úlfarsson

PhD Committee
Jóhannes R. Sveinsson
Magnús Örn Úlfarsson
Jocelyn Chanussot

Opponents
Jie Chen
Ioannis D. Schizas

Faculty of Electrical and Computer Engineering
School of Engineering and Natural Sciences
University of Iceland
Reykjavik, September 2023

Blind Hyperspectral Unmixing Using Autoencoders
Dissertation submitted in partial fulfillment of a *Philosophiae Doctor* degree in Electrical and Computer Engineering

Faculty of Electrical and Computer Engineering
School of Engineering and Natural Sciences
University of Iceland
Hjarðarhagi 2-6
107, Reykjavik
Iceland

Telephone: 525-4000

Bibliographic information:
Burkni Pálsson, 2023, *Blind Hyperspectral Unmixing Using Autoencoders*,
PhD dissertation, Faculty of Electrical and Computer Engineering, University of Iceland

Author ORCID: 0000-0001-9821-8320
ISBN: 978-9935-9742-6-6

Printing: Háskólaprent
Reykjavik, Iceland, September 2023

Abstract

The subject of this thesis is blind hyperspectral unmixing using deep learning based autoencoders. Two methods based on autoencoders are proposed and analyzed. Both methods seek to exploit the spatial correlations in the hyperspectral images to improve the performance. One by using multitask learning to simultaneously unmix a neighbourhood of pixels while the other by using a convolutional neural network autoencoder. This increases the consistency and robustness of the methods.

In addition, a review of the various autoencoder methods in the literature is given along with a detailed discussion of different types of autoencoders. The thesis concludes by a critical comparison of eleven different autoencoder based methods. Ablation experiments are performed to answer the question of why autoencoders are so effective in blind hyperspectral unmixing, and an opinion is given on what the future in autoencoder unmixing holds.

The main contributions are the following:

- An autoencoder based method (MTLAEU) which, through multitask learning, directly makes use of the spatial correlation between pixels in an HSI is introduced.

- The first fully 2D convolutional neural network autoencoder method (CNNAEU) for blind hyperspectral unmixing is introduced. Both the encoder and the decoder are 2D CNN models and the method is trained on patches from an HSI to be unmixed.

- A comprehensive review of the literature on blind hyperspectral unmixing using autoencoders. The work also gives an introduction to autoencoders and how they can be used for blind unmixing and has a critical comparison of 11 different autoencoder based methods.

Ágrip

Efni þessarar ritgerðar er aðgreining fjölrásamynda (*e. blind hyperspectral unmixing*) með sjálfkóðurum (*e. autoencoders*) byggðum á djúpum lærdómi (*e. deep learning*). Tvær aðferðir byggðar á sjálfkóðurum eru kynntar og rannsakaðar. Báðar aðferðirnar leitast við að nýta sér rúmfræðilega fylgni rófa í fjölrásamyndum til að bæta árangur aðgreiningar. Ein aðferð með að nýta sér fjölbeitingarlærdóm (*e. multitask learning*) og hin með að nota sjálfkóðara útfærðan með földunartaugnaneti (*e. convolutional neural network*). Hvortveggja bætir samkvæmni og hæfni fjölrásagreiningarinnar.

Ennfremur inniheldur ritgerðin yfirgripsmikið yfirlit yfir þær sjálfkóðaraaðferðir sem hafa verið birtar ásamt greinargóðri umræðu um mismunandi gerðir sjálfkóðara og útfærslur á þeim.

í lok ritgerðarinnar er svo að finna gagnrýninn samanburð á 11 mismunandi aðferðum byggðum á sjálfkóðurum. Brottnáms (e. ablation) tilraunir eru gerðar til að svara spurningunni hvers vegna sjálfkóðarar eru svo árangursríkir í fjölrásagreiningu og stuttlega rætt um hvað framtíðin ber í skauti sér varðandi aðgreiningu fjölrásamynda með sjálfkóðurum.

Megin framlag ritgerðarinnar er eftirfarandi:

- Ný sjálfkóðaraaðferð, MTLAEU, sem nýtir á beinan hátt rúmfræðilega fylgni rófa í fjölrásamyndum til að bæta árangur aðgreiningar. Aðferðin notar fjölbeitingarlærdóm til að aðgreina grennd af rófum í einu.

- Ný aðferð, CNNAEU, sem notar 2D földunartaugnanet fyrir bæði kóðara og afkóðara og er fyrsta birta aðferðin til að gera það. Aðferðin er þjálfuð á myndbútum (e.patches) og því er rúmfræðileg bygging myndarinnar sem greina á varðveitt í gegnum aðferðina.

- Yfirgripsmikil og ítarlegt fræðilegt yfirlit yfir birtar sjálfkóðaraaðferðir fyrir fjölrásagreiningu. Gefinn er inngangur að sjálfkóðurum og elstu tegundir sjálfkóðara eru kynntar. Gefið er greinargott yfirlit yfir helstu birtar aðferðir fyrir fjölrásagreiningu sem byggja á sjálfkóðurum og gerður er gangrýninn samburður á 11 mismunandi sjálfkóðaraaðferðum.

Acknowledgments

I am very grateful to my supervisors, professors Jóhannes and Magnús, for their guidance and support throughout this work. It has been a privilege working under their supervision. I am also grateful to Professor Jakob for good discussions and advice. I also want to express gratitude to Professor Jocelyn Chanussot for accepting to be part of my PhD committee.

I also would like to thank the opponents, Professor Jie Chen, and Associate Professor Ioannis D. Schizas for taking part in the defense and for valuable feedback and comments.

The company and friendship of other PhD students at VR-II and all the interesting discussions we had is something I am very grateful for. I wish to give my thanks to Hans, Sveinn, Bin, and Han. Thank you all for your friendship.

Finally, I want to express my deepest gratitude to my family: my parents, and my brother Frosti, for their support and encouragement. Last but not least, I give special thanks and gratitude to my good friend Fida Abu Libdeh, who has supported me greatly in many ways during these years.

This work was supported by the Icelandic Research Fund under Grants 174075-05 and 207233-051 for which I am grateful.

Contents

List of Figures

List of Tables

Acronyms

HSI Hyperspectral Image

ML Machine Learning

DL Deep Learning

HU Hyperspectral Unmixing

LMM Linear Mixing Model

NMF Nonnegative Matrix Factorization

PCA Principal Component Analysis

ASC Abundance Sum-to-One Constraint

ANC Abundance Nonnegativity Constraint

AE Autoencoder

VAE Variational Autoencoder

CNN Convolutional Neural Network

ReLU Rectified Linear Unit

LReLU Leaky Rectified Linear Unit

GAN Generative Adversarial Network

MTL Multitask Learning

Notations

$\boldsymbol{A}$	all matrices are in upper case boldsymbol
$\boldsymbol{a}_{ij}$	element in row i and column j in matrix $\boldsymbol{A}$
$\boldsymbol{x}_p$	pixel p in an hyperspectral image $\boldsymbol{X}$
$\hat{\boldsymbol{x}_p}$	reconstuction of pixel $\boldsymbol{x}_p$ by an autoencoder
$\boldsymbol{W}^{(l)}$	weight matrix of layer l in a neural network
$\boldsymbol{W}_i^{(l)}$	weight matrix of layer l, branch i
$\boldsymbol{a}^{(l)}$	activations of layer l in a neural network
g	activation function
$\boldsymbol{a}_i^{(l)}$	activations of unit i in layer l, branch i
$a_i^{(l)}$	activation of unit i in layer l
$a_{i,j}^{(l)}$	activation of unit j in $\boldsymbol{a}_i^{(l)}$
$L^{(l)}$	Number of filters/feature maps of convolutional layers l
$\mathcal{N}_k^p$	$k \times k$ neighborhood centered on pixel p

Chapter 1

Introduction

This chapter begins with introducing remote sensing hyperspectral images. The concept of hyperspectral unmixing is explained and main approaches used in unmixing are reviewed. Dimensionality reduction and its relationship to hyperspectral unmixing is also discussed. The chapter concludes with the goals and novelties of the work presented here, along with an overview of the thesis.

Over the last half-decade, the new paradigm of machine learning (ML), particularly that of deep learning (DL), has revolutionized and opened new possibilities in processing data in data-intensive fields such as remote sensing. Hyperspectral imaging, firmly a "big data" sub-field of remote sensing due to the high dimensionality of the acquired data, has not been exempt from this development. Fig. 1.1 shows the number of published papers[1] found by Web of Science[2] when using the search string "hyperspectral unmixing autoencoder". Lately, many methods developed for the processing of hyperspectral data are based on deep learning.

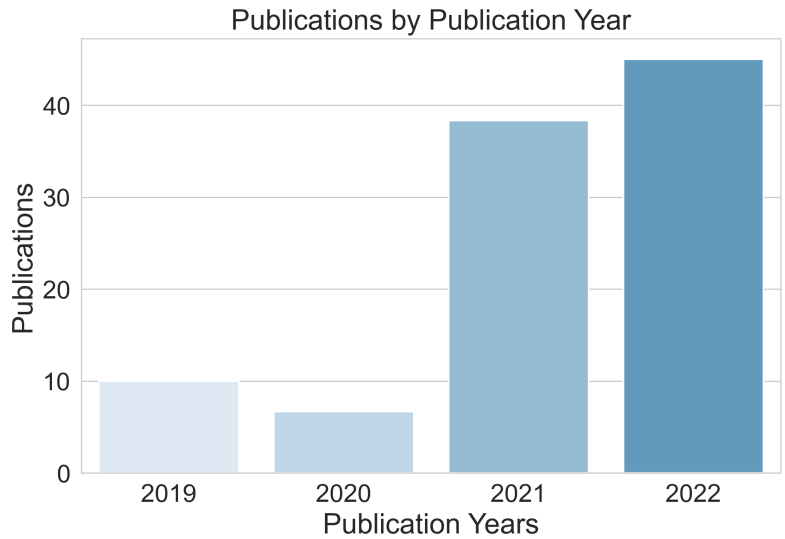


Figure 1.1: The bar plot shows the number of published papers found by Web of Science when using the search string "hyperspectral unmixing autoencoder".

[1] Search restricted to ISI journal papers.

[2] https://wcs.webofknowledge.com/RA/analyze.do?product=WOS&SID=F5DqvtI6j2uBCmrYy9m&field=PY_PublicationYear_PublicationYear_en&yearSort=true

Hyperspectral imaging belongs to the class of imaging spectrometry, where an entire spectrum is acquired at every pixel. The technique has been defined by Goetz et al., in [1] as "the acquisition of images in hundreds of contiguous, registered, spectral bands such that for each pixel a radiance spectrum can be derived". Today the number of hyperspectral imaging applications are numerous in diverse fields, including environment and earth observation, agriculture and the food industry, biotechnology, medical sciences, the pharmaceutical industry, manufacturing and forensic science [2–7].

Because of the high spectral resolution of hyperspectral images (HSIs), it is possible to determine which pure materials (endmembers) are present in a scene. However, because of the low spatial resolution of HSIs compared to multispectral images, a single pixel will often contain more than one endmember. Therefore, determining the spectra of the endmembers in an HSI and their proportions in each pixel, i.e., their abundances, is a challenging inverse problem which is the central problem of hyperspectral unmixing (HU). As a result, HU methods often determine only the endmembers, and their abundances are then subsequently determined by another method using the extracted endmembers. Methods that determine both the endmembers and their abundances simultaneously are known as blind unmixing methods.

The problem of blind spectral unmixing can be formulated as a nonnegative matrix factorization (NMF) problem, a type of blind source separation problem. Autoencoders are a type of unsupervised learning neural networks that have been known since at least 2006 [8] and are well suited for solving such problems. An autoencoder learns to reconstruct its input but is prevented from learning the identity function. By imposing a bottleneck in the network, the network is forced to discover and learn to leverage any structure present in the data, resulting in compressed knowledge representations in the bottleneck itself. Applications of autoencoders include dimensionality reduction, feature extraction, image generation, data compression, and many more [9, 10].

With the recent renaissance of neural networks, it is not surprising that autoencoders have become popular in HU research. The very first method to use an autoencoder for HU was published in 2012 and was titled "Unsupervised nonlinear spectral unmixing by means of NLPCA applied to hyperspectral imagery" [11]. This paper has gone mostly unnoticed because unsupervised NLPCA is not the keyword commonly used for autoencoders. The next method using an autoencoder for HU does not appear until 2015 [12], and now eight years later, more than one hundred papers have been published on HU using autoencoders.

1.1 Hyperspectral Unmixing

1.1.1 Hyperspectral Imaging

The history of hyperspectral imaging remote sensing is quite extensive, with its roots dating back to the 1970s. It began with field spectral measurements in support of analyzing data from NASA's Landsat-1 [13]. The development of hyperspectral imaging

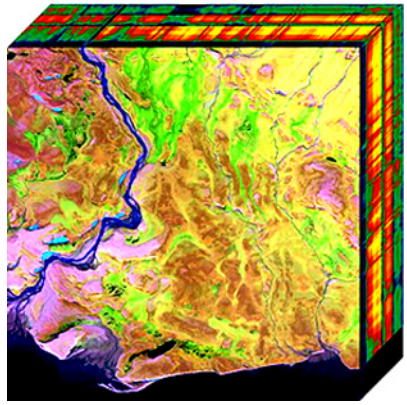


Figure 1.2: A graphical representation of an hyperspectral data cube. Source: Wikimedia Commons.

was brought about by the realization, through laboratory and field spectral measurements, mainly of minerals and soils, that multispectral imaging in four broad spectral bands with Landsat's multispectral scanner (MSS) was insufficient to distinguish and identify minerals on the earth's surface, which were crucial in resource exploration and environmental assessment [13].

In the context of the remote sensing of the earth, airborne or spaceborne hyperspectral sensors simultaneously acquire images in up to several hundred contiguous spectral bands. The sensor captures both the light emitted and reflected by the object as a spectrum consisting of several hundreds of channels, essentially a spectral response curve. The resulting images have up to hundreds of channels, compared to only a handful for traditional multispectral images, and they are organized into stacked planes forming a data cube. Fig. 1.2 shows an example of an HSI data cube.

Because the measured signal from the surface, i.e., the reflected sunlight (radiance), is affected by atmospheric effects such as clouds, water vapor and aerosols, it is converted to reflectance. Reflectance is defined as the ratio between the flux coming from the surface and the incidental flux, and is an intrinsic property of the materials being imaged. This conversion minimizes the effects of the imaging conditions. One of the challenges of hyperspectral imaging is atmospheric interference, which can affect the accuracy of the captured data. The Earth's atmosphere contains molecules and aerosols that absorb and scatter the incoming solar radiation, leading to a loss of energy in the spectrum. This results in a distortion of the hyperspectral data, making it difficult to accurately interpret the data and obtain meaningful results [14].

Atmospheric correction is the process of removing the atmospheric interference from hyperspectral images, which is essential for accurate data analysis. The atmospheric correction process involves estimating the amount of atmospheric interference in the data and then removing it to reveal the true spectral information of the surface. There

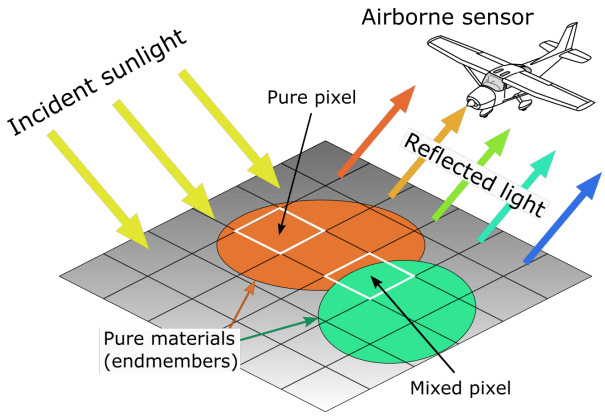


Figure 1.3: A graphical representation of hyperspectral image aquisition using an airborne senseor.

are several methods of atmospheric correction, including empirical, radiative transfer-based, and hybrid approaches. The choice of method depends on the characteristics of the hyperspectral data and the atmospheric conditions [15].

Since every pixel in an HSI corresponds to a spectral response curve, the image itself is a three-dimensional cube of either radiance or reflectance values. The reflectance spectra of pure materials are known as *endmembers*. Because of the low spatial resolution of hyperspectral sensors, there is usually more than one endmember present in a pixel, and the spectrum of the pixel is some combination of the endmembers. The proportional area that each endmember covers of the pixel is known as the *abundance fraction* of the endmember.

This is illustrated in Fig. 1.3 which shows the aquisition of an hsi using an airborne sensor. The incident sunlight is scattered off two pure materials. Some pixels, known as pure pixels, will contain only one of the materials, while others will contain both, i.e., mixed pixels. The goal of HU is to determine the endmembers and their abundance fractions in each pixel. The spectrum of a pixel can then be estimated, e.g., as a linear combination of the endmembers with the abundance fractions as the coefficients. It is evident that the abundance fractions for a pixel must sum-to-one as they are proportions. Also, the spectra of pixels are nonnegative, as they are radiance or reflectance values.

1.1.2 Mixing Models

The problem HU aims to solve arises from the fact that limited spatial resolution leads to mixed pixels, i.e., spectral mixing due to more than one pure materials within a pixel of the image. In what manner this spectral mixing happens and what assumptions can be made regarding it is central to unmixing and is captured through so-called mixing

models. There are linear and nonlinear mixing models, with the linear mixing model (LMM) being the most widely used because of its simplicity and effectiveness.

Linear Models

Linear mixing is valid when the mixing scale is macroscopic and the incident light only interacts with one pure material [16]. The spectral mixing is then actually occurring inside the sensor itself because the resolution is not fine enough to separate the materials. Under the LMM, the spectrum of a pixel $\boldsymbol{x}_p$ of an HSI $\boldsymbol{Y} \in \mathbb{R}^{w \times h \times B}$ having B bands is modelled as a convex combination of R endmembers $\boldsymbol{a}_m \in \mathbb{R}^{B \times 1}$ as

$$\boldsymbol{x}_p = \sum_{m=1}^{R} s_{mp} \boldsymbol{a}_m + \boldsymbol{\epsilon_p}, \tag{1.1}$$

where the abundance fractions s_{mp} are the area proportions of the endmembers in the pixel $\boldsymbol{x}_p$ and must satisfy the following two physically inspired constraints: $s_{mp} \geq 0$ or abundance nonnegativity constraint (ANC), $\sum_{m=1}^{R} s_{mp} = 1$ or abundance sum-to-one constraint (ASC), and $\boldsymbol{\epsilon}_p$ is noise. This can also be written as a matrix-vector multiplication as

$$\boldsymbol{x}_p = \boldsymbol{A}\boldsymbol{s}_p + \boldsymbol{\epsilon}_p, \tag{1.2}$$

where $\boldsymbol{A} \in \mathbb{R}^{B \times R}$ is the endmember matrix having the endmembers in its columns and $\boldsymbol{s}_p$ is the abundance vector. The main drawback of the LMM is its inability to represent spectral variability such as variations due to varying illumination conditions. There are other extended and augmented linear mixing models that are able to model spectral variability such as the perturbed linear mixing model in [17], the extended linear model in [18], the augmented model in [19], and the data dependent multiscale model in [20].

Nonlinear Models

When the mixing scale is not macroscopic as for intimate mixtures of materials, or the incident light scatters off multiple materials, nonlinear models are needed [21, 22]. Usually only bilinear interactions are modelled, i.e., when light reflected off one material reflects off another, a case of secondary illumination. Models that model bilinear interactions are called bilinear models [23–25].

Another case is intimate mixtures of grains or particles in close contact with each other, e.g., mineral particles in sand and soil. Light in such mixtures will typically interact multiple times with the particles making up the mixture before reaching the observer. The modelling of the optical characteristics of intimate mixtures is highly nontrivial and Hapke introduced a widely used model, the Hapke model [26].

An example of a general nonlinear model is

$$\boldsymbol{x}_p = \Theta(\boldsymbol{A}, \boldsymbol{s}_p) + \boldsymbol{\epsilon}_p, \tag{1.3}$$

where the nonlinear interactions between the endmembers in $\boldsymbol{A}$ are given implicitly by the function Θ and parameterized by the abundance vector $\boldsymbol{s}_p$. This article is only concerned with a restriction of (1.3) to the post nonlinear structure given by

$$\boldsymbol{x}_p = \Psi(\boldsymbol{A}\boldsymbol{s}_p) + \boldsymbol{\epsilon}_p, \tag{1.4}$$

where Ψ is a nonlinear function acting on the linear transform $\boldsymbol{A}\boldsymbol{s}_p$. The Hapke model can be considered a special case of (1.4). Later it will be seen that (1.4) allows for easier extraction of the endmember matrix $\boldsymbol{A}$ and the abundances $\boldsymbol{s}_p$ than the general model (1.3).

1.1.3 Spectral Unmixing Methods

Often the first step in hyperspectral unmixing is to determine how many endmembers are in a given HSI to be unmixed, which is in itself a challenging part of the unmixing problem. The most common methods for this are the method of virtual dimensionality [27], hyperspectral signal identification with minimum error (HySime) method [28], the eigenvalue likelihood maximization (ELM) method [29], and hyperspectral subspace identification using SURE [30].

Traditional methods for spectral unmixing are often grouped into three main categories, with the categories defined by how the problem is interpreted [31]. The categories are sparse regression methods, geometrical methods, and statistical methods.

Sparse regression methods

Among the earliest methods are the sparse regression methods that seek to express the observed spectra as linear combinations of known spectral signatures from spectral libraries [32–35]. These methods can be used when library of potential spectral signatures for the sensor used is available. Often these endmembers are collected from the actual location of the imagery being analyzed using field-sensors or they can be obtained from a laboratory. The spectral signatures of the endmembers are then used to construct a spectral library. The spectral library is then used to construct a dictionary matrix $\boldsymbol{D}$ whose columns are the spectral signatures of the endmembers. The abundance vector $\boldsymbol{s}_p$ is then sought as the solution to the following optimization problem [31]

$$\boldsymbol{s}_p = \min_{\boldsymbol{s}_p} \|\boldsymbol{s}_p\|_0 \text{ subject to } \|\boldsymbol{x}_p - \boldsymbol{D}\boldsymbol{s}_p\|_2 \leq \delta, \tag{1.5}$$

where $\|\cdot\|_0$ is the ℓ_0 norm and δ is a threshold. The ℓ_0 norm is the number of non-zero elements in the vector which we want to be low.

Geometrical methods

Another category of methods is geometrical methods. These methods are centered around the observation that the spectral vectors generated according to the LMM lie in a $R-1$ simplex in $\mathbb{R}^B$ with the endmembers at the vertices. Geometrical methods can be further categorized into pure pixel methods or minimum volume simplex methods based on whether they rely on the presence of pure pixels, i.e., the spectra of pure materials in the data, or not.

Making the assumption that every pure material in a scene has at least one pure pixel makes the problem very computationally efficient. A very well known and widely used pure pixel method is vertex component analysis (VCA) [36]. Another widely used pure pixel method is the N-FINDR method [37].

Minimum volume approaches do not make any assumptions about the presence of pure pixels which in turn makes the problem nonconvex and more computationally expensive. Minimum volume methods attempt to find a set of endmembers that minimize the volume of the simplex defined by the endmembers subject to the constraint that the observed data must fit within the simplex [31].

A necessary condition for convergence of these methods is the presence of a certain minimum number of pixels on each facet of the simplex, a condition that may not hold for highly mixed data [31]. A good exmaple of a minimum volume technique is minimum volume simplex analysis (MVSA) [38]. Further examples of $\ell_{1/2}$ and ℓ_q sparsity constrained minimum volume methods are [39] and [40], respectively.

Statistical methods

Statistical methods reformulate the unmixing problem as an inference problem. In blind unmixing, where we want to simultaneously determine the endmembers and their fractional abundances, the problem becomes a blind source separation problem (BSS) [41–43]. Because of the nonnegativity constraint in mixing models, nonnegative matrix factorization (NMF) has been widely used by blind unmixing methods [39, 40, 44]. Two widely used NMF methods are $\ell_{1/2}$-NMF [39] and MVC-NMF [45]. Most statistical approaches to unmixing are either variants or extensions of NMF. Methods based on compressed sensing, such as [46–48], also belong to this category of statistical methods.

1.2 Notation

In this thesis the following notation will be used throughout: The number of pixels in an HSI $\boldsymbol{X}$ will be denoted with the letter P and the number of bands with the letter B. The letter R will be used to denote the number of endmembers to be estimated. The endmember matrix will be denoted $\boldsymbol{A}$ and will be of size $B \times R$, and endmember

i will be denoted with a_i, a vector of size $B \times 1$. The abundance vector $\boldsymbol{s}_p$ of pixel p will be of size $R \times 1$. The observed spectra $\boldsymbol{x}_p$ will be of size $B \times 1$. The $k \times k$ neighborhood of pixels centered on pixel p will be denoted with $\mathcal{N}_k^p$.

When discussing neural networks the following notation will be used: The weights of layer number l will be denoted with the letter $\boldsymbol{W}^{(l)}$. If the neural network has branches, the weight matrix of layer l and branch i will be denoted with $\boldsymbol{W}_i^{(l)}$. The vector of activations of layer l will be denoted with $\boldsymbol{a}^{(l)}$ and a subscript will be used to indicate branch number if the neural network has branches. The activation of unit i in layer l will be denoted $a_i^{(l)}$, and another subscript will be added if the layer is a branch.

The activation function of a neural network will be denoted with the letter g and batch normalization with the letters BN. The network's loss function will generally be denoted with $\mathcal{L}$ and fidelity measures, and sometimes regularization terms, with the letter J.

1.3 HSIS USED IN EXPERIMENTS AND PERFORMANCE MEASURES

1.3.1 DATASETS

All experiments were performed using four real HSIs described below and four synthetic datasets with increasing spectral variability. The methodology for determining the reference endmembers is described in [49]. The datasets are:

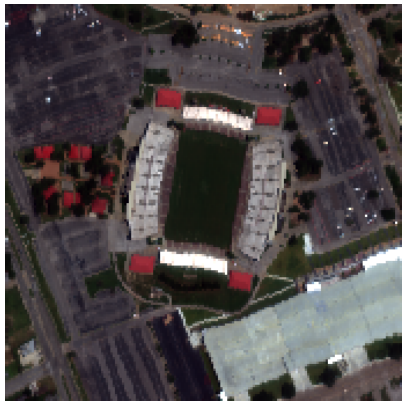
1. **Samson.** The SAMSON sensor obtained this widely used dataset, and it is a cropped image from a larger image. The size is 95 by 95 pixels, and the number of bands is 145 covering the 401-889 nm wavelength range. The dataset has the following endmembers: Water, Soil, and Tree.

2. **Urban.** Obtained by the HYDICE [50] (Hyperspectral Digital Image Collection Experiment) sensor, this image is 307 by 307 pixels and has 210 bands covering the 400-2500 nm wavelength range. After removing corrupted and noisy bands, 162 bands remain. Here, 4, 5 and 6 reference endmembers are used in experiments. Grass, Tree, Asphalt, and Roof were selected as the references for four endmembers. The five endmembers reference additionally includes the Soil endmember, and the six reference endmembers reference additionally includes the Soil and Metal endmembers.

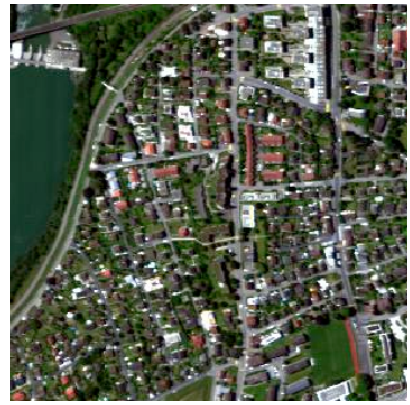
3. **Houston.** This dataset is a cropped image from a larger one acquired over the University of Houston campus, Texas, USA, in June 2012, and has 144 bands covering the wavelengths of 380-1050 nm with a spatial resolution of 2.5 meters. The cropped image is 170 by 170 and is centred on the Robertson Stadium on

(a) Samson - simulated

(b) Urban - simulated



(c) Houston - actual



(d) Apex - actual

Figure 1.4: Simulated and actual RGB images of the datasets used in the experiments.

the Houston Campus and surrounding area. For this dataset, four endmembers are estimated, and the reference endmembers are selected from the averages of the 15 classification ground truth categories that come with the dataset.

4. **Apex.** This hyperspectral dataset is a cropped image from a much larger one acquired by the APEX [51] sensor during a clear day in June 2011 at an altitude of 4600 m above sea level with a heading of 56 degrees in the vicinity of Baden, Switzerland. Two hundred eighty-five bands cover the wavelength range between 413 nm and 2412 nm, all usable. The cropped image is 300 by 300 pixels sub-image cropped from the larger 1500×1000 image at location (70,650). Here four endmembers are estimated, and the reference endmembers are Asphalt, Vegetation, Water, and Roof.

1.3.2 Performance measures

The endmembers extracted by the methods are quantitatively evaluated by calculating the average SAD from the reference endmembers using

$$\text{mSAD} = \frac{1}{R}\sum_{i=1}^{R}\cos^{-1}\left(\frac{\langle \hat{\boldsymbol{m}}_i, \boldsymbol{m}_i\rangle}{\|\hat{\boldsymbol{m}}_i\|_2 \|\boldsymbol{m}_i\|_2}\right), \tag{1.6}$$

where $\hat{\boldsymbol{m}}_i$ are the endmembers extracted by a method, and $\boldsymbol{m}_i$ are the reference endmembers. The lower the SAD, the better the estimated endmembers resemble the reference endmembers. In result tables, the SAD for individual endmembers will also be given. The extracted abundance maps are also quantitatively evaluate for each dataset by calculating the average mean squared error between them and the reference abundance maps using

$$\text{MSE} = \frac{1}{P}\sum_{i=1}^{P}\|\mathbf{S}_i - \hat{\mathbf{S}}_i\|^2, \tag{1.7}$$

where $\hat{\mathbf{S}}_i$ are the abundance values for a particular endmember for pixel i and $\mathbf{S}_i$ are the reference abundance values.

1.4 Thesis contributions and organization

The topic of this thesis is blind hyperspectral unmixing using autoencoders with special focus on the incorporation of spatial information, i.e., to exploit the spatial structure of HSIs in addition to their spectral information. The pixels in HSIs are, as in other natural images, strongly correlated with their neighbors. This spatial structure contains information that can be used along with the spectral information to constrain further the unmixing problem.

Unmixing methods that exploit the spatial structure are called spectral-spatial methods. The most common way to utilize the spatial structure of HSIs is by refining a spectral method by using some spatial prior to regularize or control the sparsity and the smoothness of the abundance maps obtained [52] or by doing preprocessing on the data prior to using a spectral only method. Developing entirely new spatially oriented methods is not as common.

Chapter 2 is an introduction to autoencoders and how autoencoders are used for spectral unmixing. This chapter will discuss the details of implementing such autoencoders, such as choices of activation functions, different ways of implementing the ASC and ANC constraints, and the effects of different objective functions.

In Chapter 3, an autoencoder based method that uses multitask learning to simultaneously unmix a neighborhood of pixels will be introduced. This method thus attempts to directly exploit the spatial structure of the HSI by utilizing an encoder that has multiple branches or tasks, that share a layer, while using a single shared decoder. It

will be demonstrated that this improves the consistency of the method and lowers the variance of estimated endmembers.

In Chapter 4, the first fully convolutional autoencoder for spectral unmixing will be introduced. The technique exploits the spatial and the spectral structure of HSIs both for endmember and abundance map estimation. As it works directly with patches of HSIs and does not use any pooling or upsampling layers, the spatial structure is preserved throughout, and abundance maps are obtained as feature maps of a hidden convolutional layer. It will be demonstrated that this improves the performance and consistency of the method compared with strictly spectral unmixing autoencoder methods.

In Chapter 5, a comprehensive comparison and evaluation of many autoencoder based method from the literature will be given. Eleven different, including the author's own, autoencoder based methods will be compared using four real hyperspectral datasets.

1.5 Publications

Chapter 2 is based on the following publications:

[a] Palsson, B., Sveinsson, J. & Ulfarsson, M. Blind Hyperspectral Unmixing Using Autoencoders: A Critical Comparison. *IEEE Journal Of Selected Topics In Applied Earth Observations And Remote Sensing.* **15** pp. 1340-1372 (2022)

Chapter 3 is based on the following publications:

[b] Palsson, B., Sveinsson, J. & Ulfarsson, M. Multitask Learning for Spatial-Spectral Hyperspectral Unmixing. *IGARSS 2019 - 2019 IEEE International Geoscience And Remote Sensing Symposium.* pp. 564-567 (2019)

[c] Palsson, B., Sveinsson, J. & Ulfarsson, M. Spectral-Spatial Hyperspectral Unmixing Using Multitask Learning. *IEEE Access.* **7** pp. 148861-148872 (2019)

Chapter 4 is based on the following publications:

[d] Palsson, B., Ulfarsson, M. & Sveinsson, J. Convolutional Autoencoder for Spatial-Spectral Hyperspectral Unmixing. *IGARSS 2019 - 2019 IEEE International Geoscience And Remote Sensing Symposium.* pp. 357-360 (2019)

[e] Palsson, B., Ulfarsson, M. & Sveinsson, J. Convolutional Autoencoder for Spectral-Spatial Hyperspectral Unmixing. *IEEE Transactions On Geoscience And Remote Sensing.* pp. 1-15 (2020)

Chapter 5 is based on the following publications:

[a] Palsson, B., Sveinsson, J. & Ulfarsson, M. Blind Hyperspectral Unmixing Using Autoencoders: A Critical Comparison. *IEEE Journal Of Selected Topics In Applied Earth Observations And Remote Sensing.* **15** pp. 1340-1372 (2022)

Chapter 2

Autoencoders

This chapter discusses how autoencoders are applied to blind hyperspectral image unmixing. The chapter starts with a brief introduction to autoencoders, and then discusses the main types of autoencoders that have been used in the literature for unmixing. Various ways to implement the necessary constraints of the LMM are also discussed in detail. Next, various architectural choices are discussed, along with common regularizations used in the literature and how they are implemented. Also discussed is how to incorporate spatial information into the autoencoder framework and the proplem of hyperparameter selection and optimization.Next is short discussion on what makes autoencoders so effective for unmixing. The chapter conludes with a review of non-blind, i.e., abundance maps estimation methods using autoencoders.

Fig. 2.1 shows a schematic of an autoencoder. An autoencoder consists of two parts,

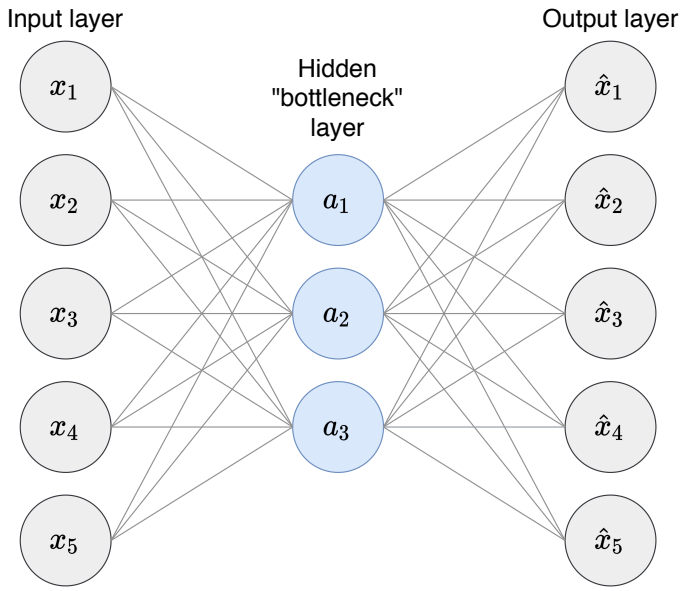


Figure 2.1: A schematic of a simple autoencoder with a single hidden layer bottleneck.

an encoder and a decoder. The encoder, $\mathcal{G}_E$, encodes the input, $\boldsymbol{x}_p$, into a latent code or a hidden representation, $\boldsymbol{h}_p = \mathcal{G}_E(\boldsymbol{x}_p)$, in a latent space of typically much lower dimension. The decoder, $\mathcal{G}_D$, must then reconstruct the original input from the latent

code, $\hat{\boldsymbol{x}}_p = \mathcal{G}_D(\boldsymbol{h}_p)$, to minimize the loss

$$\mathcal{L}(\boldsymbol{x}_p, \mathcal{G}_D(\mathcal{G}_E(\boldsymbol{x}_p))), \tag{2.1}$$

where $\mathcal{L}(\boldsymbol{x}_p, \hat{\boldsymbol{x}_p})$ is some measure of the discrepancy between the original input $\boldsymbol{x}_p$, and the reconstruction by the autoencoder, $\hat{\boldsymbol{x}}_p$. If $\boldsymbol{W}_E$ denotes the weights matrix of the encoder, and $\boldsymbol{W}_D$ denotes the weights matrix of the decoder, the forward pass of the simple autoencoder in Fig. 2.1 can be written as

$$\hat{\boldsymbol{x}}_p = g_D(\boldsymbol{W}_D(g_E(\boldsymbol{W}_E\boldsymbol{x}_p))), \tag{2.2}$$

where g_E and g_D are the activation functions of the hidden layer and the output layer, respectively. The reconstruction loss is indifferent to latent space characteristics, so generally we need some additional constraints to ensure the autoencoder learns meaningful representations, i.e., that it learns the data manifold.

Thus, by using appropriate regularization, we can force the autoencoder to only learn the necessary variations to reconstruct training examples. This balance between being sensitive enough to the inputs to reconstruct them accurately, and being insensitive enough to them to learn the actual data distribution, is essential in regularized autoencoders. The regularized loss has the form

$$\mathcal{L}_{\text{regularized}}(\boldsymbol{x}_p, \hat{\boldsymbol{x}}_p) = \mathcal{L}(\boldsymbol{x}_p, \hat{\boldsymbol{x}}_p) + \lambda J(\boldsymbol{h}_p, \boldsymbol{W}_E, \boldsymbol{W}_D), \tag{2.3}$$

where $J(\boldsymbol{h}_p, \boldsymbol{W}_E, \boldsymbol{W}_D)$ is some penalty that can be a function of the activities of the units in the bottleneck layer, the weights of the decoder, or the encoder, and the lambda is the tuning parameter which controls how strong the regularization is.

Now, if we wish to perform spectral unmixing using an autoencoder such as in Fig. 2.1, it is easy to derive the basic architecture of the autoencoder by comparing the forward pass given by (2.2) to the LMM given by (1.2). The activation function of the decoder, g_E, is required to be linear, then $\boldsymbol{A} = \boldsymbol{W}_D$ and $\boldsymbol{s}_p = g_E(\boldsymbol{W}_E\boldsymbol{x}_p)$.

The encoder encodes the input spectrum to a latent code that can be interpreted as the abundance fractions. The linear decoder layer then reconstructs the input by creating a linear combination of the columns of its weights matrix (the endmembers) with the abundance fractions as the coefficients. The simple autoencoder, however, fails to satisfy the ANC and ASC constraints. Hence, a bit more sophisticated architecture is needed to implement the LMM fully.

The following few sections will give a brief overview of the different types of autoencoders that have been used for hyperspectral unmixing.

2.1 Sparse Nonnegative Autoencoders

It is possible to make autoencoders discover and utilize structures in the data without having a bottleneck layer with fewer units than the input layer. This can be done by forcing the activations in the hidden layers to be sparse, again creating an information bottleneck that prevents the autoencoder from just learning the identity function. Forcing sparse activations in the hidden layers means that we encourage the network to learn to encode and decode a given input in a manner that only activates a few hidden units at a time in every hidden layer.

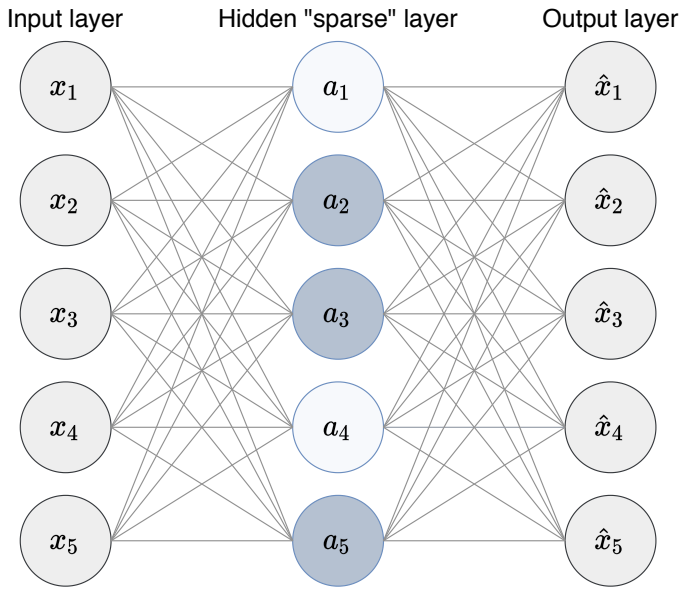


Figure 2.2: A schematic of a sparse autoencoder with a single hidden layer. Dark hidden units have higher activation strength than the lighter ones.

However, since the number of bands in an HSI is so much greater than the number of endmembers to estimate, sparse autoencoders for unmixing use a *sparse bottleneck* layer, i.e., a layer that has much fewer units than the input layer, and which is required to be sparse via a regularizing term in the loss function such as ℓ_1-norm penalty on the activities of the units in the bottleneck layer.

In order to satisfy the ANC constraint, the abundance fractions, i.e., the output of the encoder part, must be nonnegative. This can be done in more than one way. One way is to select an activation function for the last encoder layer that outputs only nonnegative numbers such as the rectified linear unit activation (ReLU) [53] given by

$$\mathrm{ReLU}(z_i) = \max(0, z_i), \tag{2.4}$$

or the sigmoid activation function, given by

$$\text{sigmoid}(z_i) = \frac{e^{z_i}}{1 + e^{z_i}}. \tag{2.5}$$

Another way is to let the implementation of the ASC constraint also take care of the ANC constraint. Using, e.g., the softmax activation function,

$$\sigma(\boldsymbol{z})_i = \frac{e^{z_i}}{1 + \sum_{j=1}^{R} e^{z_j}}, \tag{2.6}$$

to implement the ASC constraint both ensures the abundance fractions sum-to-one and that they are nonnegative. Later in this article, different ways to implement the ASC constraint will be discussed.

It is also necessary to make sure that the weights of the decoder are all nonnegative as they correspond to the endmembers, which represent reflectance values and must therefore be nonnegative. This can be done in two ways: employing a nonnegativity kernel constraint in the deep learning framework used to implement the method or using regularization that penalizes negative weights. The sparse autoencoder that satisfies all of these nonnegativity constraints is known as a sparse nonnegative autoencoder. Such an autoencoder is effectively implementing NMF. It factors the observed HSI into the product of two nonnegative matrices, namely the nonnegative matrix of abundance fractions and the mixing matrix having the endmembers as its columns. Such autoencoders are widely used in problems that can be solved through NMF, such as various blind separation problems [54, 55].

2.2 Variational Autoencoders

The latent space of traditional autoencoders is very irregular, which makes them unsuitable as generative models [56]. Here, irregular means that two points close in latent space are not guaranteed to be similar once decoded, or put another way; two similar inputs are not necessarily encoded to points close together in the latent space. Variational autoencoders (VAE) [57] solve this problem by utilizing specialized regularized training that ensures that the latent space has good properties. Fig. 2.3 illustrates the essential difference between standard and variational autoencoders. In a variational autoencoder, the encoder encodes the input to distributions over latent space instead of values. This is done by having the encoder model output the parameters of distributions and then sampling from these distributions the input values for the decoder model. VAEs make a strong assumption about the true prior distribution, $p(\boldsymbol{z}|\boldsymbol{x})$, of the latent space representations. During training, a loss term is added to the objective function that makes the learned distribution $q(\boldsymbol{z}|\boldsymbol{x})$ similar to the assumed true prior distribution $p(\boldsymbol{z}|\boldsymbol{x})$. The general loss function of a VAE is given by

$$\mathcal{L}_{\text{VAE}}(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \mathcal{L}(\boldsymbol{x}, \hat{\boldsymbol{x}}) + \sum_j KL(q_j(\boldsymbol{z}, \boldsymbol{x})||p(\boldsymbol{z})), \tag{2.7}$$

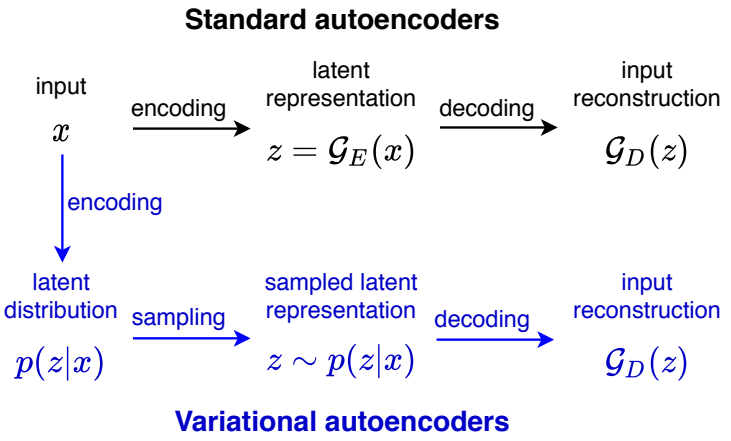


FIGURE 2.3: A comparison between standard autoencoders and variational autoencoders (blue path). Variational autoencoders encode the input into a probability distribution over latent space. The decoder samples from this distribution when decoding the encoding.

where KL is the Kullback-Leibler divergence [58] and the sum is over the dimensions of the latent space [57].

2.3 ADVERSARIAL AUTOENCODERS

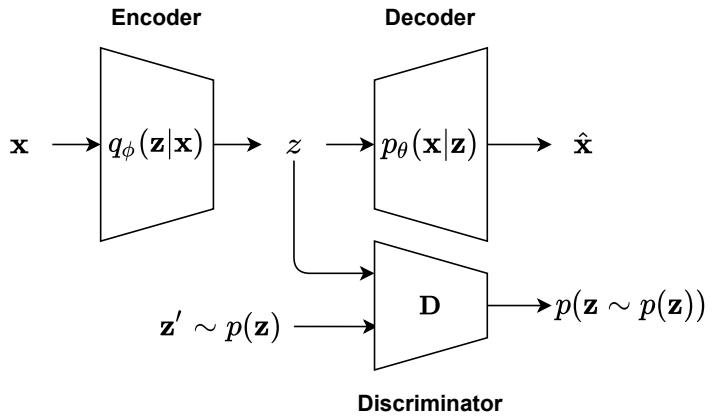


FIGURE 2.4: A schematic of an adversarial autoencoder.

An adversarial autoencoder is an approach that borrows ideas from generative adversarial networks (GAN) to turn a regular autoencoder into a useful generative model [59]. A schematic of an adversarial autoencoder is shown in Fig. 2.4. The

upper part is a standard autoencoder that tries to reconstruct its input $\boldsymbol{x}$. It has a encoder model, $q_\phi(\boldsymbol{z}|\boldsymbol{x})$, that produces a latent code $\boldsymbol{z}$, and a decoder model, $p_\theta(\boldsymbol{x}|\boldsymbol{z})$, that reconstructs the input from the latent code. The encoder model has an aggregated posterior $q(\boldsymbol{x})$ such that $z \sim q(\boldsymbol{x})$, that is given by

$$q(\boldsymbol{x}) = \int_x q_\phi(\boldsymbol{z}|\boldsymbol{x}) p_d(\boldsymbol{z}) d\boldsymbol{x}, \tag{2.8}$$

where $p_d(\boldsymbol{z})$ is the distribution of the data.

An adversarial autoencoder is obtained by regularizing the autoencoder described above by matching the aggregated posterior $q(\boldsymbol{x})$ to some arbitrary prior distribution $p(\boldsymbol{z})$ [59]. This is done by borrowing the concept of an adversarial network from GANs. The network labeled D in Fig. 2.4 is trained to tell if the latent code comes from the aggregate posterior $q(\boldsymbol{x})$ or from the prior distribution $p(\boldsymbol{z})$. The autoencoder's encoder acts as a generator of an adversarial network while the network D acts as a discriminator.

During training, the adversarial network and the autoencoder are trained alternately. First, the autoencoder updates both the encoder and the decoder to minimize the reconstruction error. Then the adversarial network updates its discriminator network D to tell apart the true samples (sampled from the prior $p(\boldsymbol{z})$) from the ones generated by the encoder part and updates its generator (the encoder) to confuse the discriminator D.

2.4 Denoising Autoencoders

Autoencoders having only a single hidden layer and which take a partially corrupted input and are trained to recover the original uncorrupted input are known as denoising autoencoders (DA) [60]. At training time, the input $\boldsymbol{x}$ is partially corrupted using a stochastic mapping, resulting in $\tilde{\boldsymbol{x}}$. The partially corrupted input is then reconstructed in the standard way, $\hat{\boldsymbol{x}} = \mathcal{G}_D(\mathcal{G}_E(\tilde{\boldsymbol{x}}))$, but the loss is calculated between the reconstructed input $\hat{\boldsymbol{x}}$ and the original uncorrupted input, $\boldsymbol{x}$ [60]. It is important to notice that the input is only degraded during training. To perform the denoising well, the model needs to extract features that capture useful structure in the input distribution.

2.5 Marginalized Denoising Autoencoders

Many simple denoising autoencoders are often stacked to form a deep network by feeding the latent code of the previous DA as input to the current DA in the stack [60]. The DAs are pre-trained one layer at a time, with each layer trained as a denoising autoencoder by minimizing the error in reconstructing its input which is the output latent code of the previous layer. When stacked denoising autoencoders (SDAs) are used for feature learning, linear autoencoders are sufficient, i.e., the activation function

can be set to be linear, and the effect of each autoencoder can be given by a single linear transformation $\boldsymbol{W}$. When considering a deep enough stack of such single layer linear denoisers, the corruption can be marginalized out, and a closed-form solution can be obtained for the effective reconstructing matrix $\boldsymbol{W}$ [61]. The resulting single-layer autoencoder, which applies the reconstructing matrix, is known as a marginalized denoising autoencoder (mDA) and has found use in spectral unmixing [61, 62].

2.6 Convolutional Autoencoders

Convolutional autoencoders [63] are based on convolutional neural networks (CNNs). These networks use convolutional layers instead of fully connected layers, which use convolutions of learnable filters with input patches to produce feature maps. In the context of spectral unmixing, they differ from conventional autoencoders in one crucial aspect, which is that they are inherently spatial and make use of the spatial correlations existing in HSIs. In contrast, conventional autoencoders look at one spectrum at a time and need special regularization to consider spatial information.

Architecturally, they are similar to conventional autoencoders used in spectral unmixing. For example, the abundance maps arise naturally as the feature maps of a "spectral bottleneck" layer, i.e., a convolutional layer having R feature maps, one for each endmember. Similarly, the weights of the linear decoder convolutional layer can be interpreted in terms of endmembers, but to do so for the non-trivial case of filter size larger than 1×1 requires a new spectral-spatial linear mixing model as the reconstruction of a pixel's spectrum involves the abundances of neighbouring pixels [64].

2.7 Spectral Unmixing Autoencoders

The majority of autoencoder based methods for blind unmixing are based on the fully connected sparse nonnegative autoencoder. Here we will take a closer look at various implementation details and choices taken by methods in blind unmixing that can influence the performance of spectral unmixing, e.g., choice of loss functions, choice of activation functions, and so forth. Fig. 2.5 shows a generic autoencoder that performs spectral unmixing.

It has a deep encoder with two hidden layers, but the hidden layers can vary from one to several. The decoder has a nonnegative constraint for its weights and linear activation. It is also possible to make the decoder's weights nonnegative using a regularization that penalizes negative weights. The activation function of the encoder is ReLU or Leaky ReLU LReLU). The ASC constraint is enforced using a normalizing utility layer or softmax activation. In practice, it is convenient to create a custom layer that enforces the ASC because it makes it easier to implement abundance regularizations. The forward pass of this general autoencoder can be written out as

$$\hat{\boldsymbol{x}}_p = \boldsymbol{W}_D(\boldsymbol{\sigma}\,\mathrm{BN}\, g(\boldsymbol{W}^{(2)}\mathrm{BN}\, g(\boldsymbol{W}^{(1)}\boldsymbol{a}^{(0)}))), \tag{2.9}$$

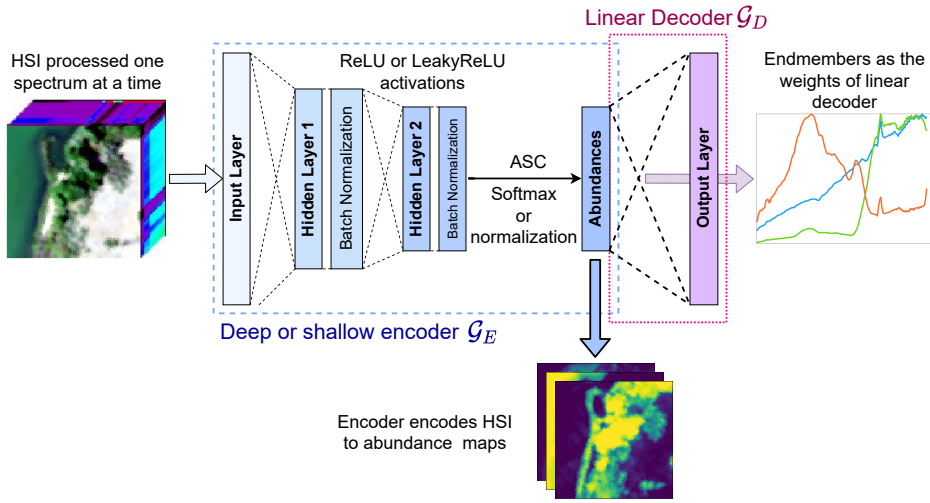


Figure 2.5: A schematic of a general unmixing autoencoder. The figure shows an example having two hidden layers but the actual number could vary from one to several. The decoder has a nonnegative weights constraint and linear activation. The ASC constraint is enforced with a normalizing utility layer or softmax activation.

where BN denotes batch normalization using operator notation, and $\boldsymbol{\sigma}$ is similar operator notation for the action of the ASC enforcing layer, i.e., $\boldsymbol{\sigma a}$ sums-to-one, $\boldsymbol{a}^{(0)} = \boldsymbol{x}_p$, i.e., the input layer activations, and $\boldsymbol{W}_D$ is the weight matrix of the linear decoder.

2.7.1 Deep versus shallow encoder

The power of depth in feedforward neural networks is still not fully understood in the field of deep learning theory [65]. Deep neural networks can extract better features than shallow networks for the same amount of computation. In [65], it is proven that deep networks can approximate the class of compositional functions with the same accuracy as shallow networks but with an exponentially lower number of training parameters. However, deep networks are more prone to overfitting and can be harder to train than shallow networks and generally require more regularizations. Whether a deep encoder will extract better abundance fractions than a shallow encoder most likely will depend on the mixing model being assumed. It could be argued that a deep encoder can extract better abundances under a nonlinear mixing model. We will come back to this question when discussing the experimental results. The methods in [64, 66–71] employ deep encoders.

2.7.2 Batch normalization

Batch normalization is used to speed up the training of neural networks and make them more stable through normalization of the inputs to a layer by re-centring and re-scaling [72]. Initially, it was believed that batch normalization did work by mitigating internal covariance shift, which reduces the learning rate of the network [72], but recent research shows that it works by smoothing the objective function [73]. Batch normalization layers are usually used after or before a nonlinear activation function such as ReLU, and they are only active during training.

Regarding the choice of batch size, it is the authors' experience that small batch sizes yield better unmixing performance than large batch sizes. In [74], it is argued that large batch sizes can lead to degradation of the quality of deep learning models as measured by their generalization abilities as large-batch methods tend to converge to sharp minimizers of the training and testing functions, and sharp minima lead to more inferior generalization.

If we let $\boldsymbol{a}_i$, $i = 1, \ldots, m$ 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

$$\boldsymbol{a}_{\mathrm{BN}_i} = \mathrm{BN}_{\gamma,\beta}(\boldsymbol{a}_i) = \gamma \hat{\boldsymbol{a}}_i + \beta, \tag{2.10}$$

where

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

γ and β are learnable parameters, and ϵ is a very small number.

2.7.3 Choice of activation function for hidden layers

The choice of activation function for the hidden layers and especially the last hidden layer of a deep encoder can significantly influence the behaviour and performance of unmixing autoencoders. Like in deep learning in general, the trend has been away from using saturating activation functions which squeeze their input, e.g., like the sigmoid function, to using non-saturating activation functions such as the Leaky ReLU (LReLU) variant [75] given by

$$\mathrm{LReLU}(x) = \max(\beta x, x), \tag{2.11}$$

where β is a real number between 0 and 1. The problem with saturating activation functions is that the value of hidden units often becomes stuck at their asymptotic values, e.g., 0 or 1 for the sigmoid activation, indicating that the pre-activation sum-of-products is relatively large. This leads to vanishing gradients during training. The methods in [12, 76] use a parameterized sigmoid activation function. The ReLU activation largely mitigates the problem of vanishing gradients, but the function is still saturated to the left, and hidden units can become stuck at zero. In unmixing, this can cause the abundance maps of certain endmembers to become zero-valued. The works [62, 70, 71, 77, 78] all use the ReLU activation. The LReLU activation is a good choice of an activation function as it is non saturating in both directions. The methods in [64, 67, 69] all use LReLU as the activation function of hidden units.

2.7.4 Dropout

Dropout is a powerful regularization technique that helps with preventing overfitting and improving the generalization of deep networks. It works by randomly omitting or dropping units of a layer during the training of a neural network [79]. This prevents complex co-adaptations of units on the training data. It can be thought of as a way of performing model averaging with neural networks. In convolutional networks, feature maps, instead of units, are set randomly to zero or dropped, a case of dropout known as spatial dropout [80]. When properly used, dropout can give a better unmixing performance, primarily when used with deep encoders as they need more regularization. Dropout is not commonly used in blind unmixing methods. The methods in [66, 67, 77] use dropout while the method in [64], being a convolutional network, uses spatial dropout.

2.7.5 Choices of loss fidelity function and spectral variability

The choice of function or similarity measure used for the fidelity term of the loss function is a crucial one that can significantly influence the performance of the unmixing autoencoder. The main reason for this is that some similarity measures are scale-invariant while others are not. Furthermore, due to variations in illumination or topography, real hyperspectral data often has considerable spectral variability and is not well modelled by the LMM. This means that the same material or mixture of materials can have spectra that differ in scale even though they have the same abundances.

A similarity measure that is not scale-invariant will penalize the LMM reconstructions that differ in scale from the original spectra and thus cannot handle spectral (scale) variability. In contrast, a scale-invariant similarity measure will only penalize based on the difference in shape. Using a scale sensitive similarity measure for an HSI that exhibits substantial spectral variability leads to worse unmixing performance than if a scale-invariant measure had been used. The most common scale-invariant similarity

measures used in unmixing are the spectral angle distance (SAD) given by

$$J_{\text{SAD}}(\boldsymbol{x}_p, \hat{\boldsymbol{x}}_p) = \arccos\left(\frac{\langle \boldsymbol{x}_p, \hat{\boldsymbol{x}}_p \rangle}{\|\boldsymbol{x}_p\|_2 \, \|\hat{\boldsymbol{x}}_p\|_2}\right), \tag{2.12}$$

and the spectral information divergence (SID) measure given by

$$J_{\text{SID}}(\boldsymbol{x}_p, \hat{\boldsymbol{x}}_p) = \sum_{n=1}^{B} p_n \log\left(\frac{p_n}{q_n}\right) + \sum_{n=1}^{B} q_n \log\left(\frac{q_n}{p_n}\right), \tag{2.13}$$

where

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

are estimates of the probability mass functions of the target and estimated spectra, respectively. The methods in [64, 66, 67, 69] all use the SAD similarity measure as the fidelity term of the objective function, while the method in [81] uses the SID measure. The most common non-scale invariant similarity measure is the mean squared error (MSE) given by

$$J_{\text{MSE}}(\boldsymbol{x}_p, \hat{\boldsymbol{x}}_p) = \|\boldsymbol{x}_p - \hat{\boldsymbol{x}}_p\|_2^2. \tag{2.14}$$

All the methods in [12, 62, 68, 76, 78, 82–84] use the MSE measure as the fidelity term of the network's loss function. It is also possible to use a combination of both scale-invariant and non-scale-invariant loss terms. The work in [77] uses such a combination as the fidelity term. In [70], it is argued that the histogram of the reconstruction error of an unmixing autoencoder follows a hyper-Laplacian distribution and hence an optimal fidelity term should have the form

$$J_{\text{HL}}(\boldsymbol{x}_p, \hat{\boldsymbol{x}}_p) = \|\boldsymbol{x}_p - \hat{\boldsymbol{x}}_p\|_q^q, \tag{2.15}$$

where q is a positive real number.

2.7.6 Implementation of the ASC constraint

The last step the encoder needs to do is to make sure the latent code resulting from the nonlinear dimensionality reduction of previous layers sums-to-one in accordance with the ASC constraint of the LMM. One way to do this is to normalize the latent code vector, $\boldsymbol{s}$, resulting from the last layer of the encoder as

$$\hat{s}_i = \frac{s_i}{\sum_{j=1}^{R} s_j}, \tag{2.16}$$

where $\hat{s}$ is the abundance vector. If this method is to be used, it must be ensured that all the values $s_i, i = 1, \cdots, R$ are nonnegative. In practice, this means that the choice of activation function for the last layer of the encoder is restricted to strictly nonnegative functions such as ReLU or sigmoid activation functions. It might be

argued that thresholding the values using ReLU or dynamic ReLU,

$$\hat{\boldsymbol{s}} = \max(0, \boldsymbol{s} - \boldsymbol{\alpha}), \tag{2.17}$$

where α is a nonnegative $R \times 1$-vector learned by the network, as some works do, might deactivate many units in the network and not utilize its full capacity. The methods in [12, 66] use this implementation of the ASC, while [66] additionally uses it with a dynamically thresholding ReLU activation. Another ASC implementation that does not require strictly nonnegative activation of the last hidden layer is

$$\hat{s}_i = \frac{|s_i|}{\sum_{j=1}^{R} |s_j|}. \tag{2.18}$$

This implementation also takes care of the ANC constraint. The method in [69] uses this implementation. The method described in [77] uses a variation of (2.18), where taking the absolute value in the nominator is omitted. Yet another implementation of the ASC that many works use, is to apply the softmax function directly to $\boldsymbol{s}$ or scale it first as in

$$\hat{\boldsymbol{s}} = \text{softmax}(\gamma \boldsymbol{s}), \tag{2.19}$$

where γ is a constant larger than one. This implementation also takes care of the ANC constraint and by choosing large γ achieves something similar to ℓ_1-norm regularization on the abundance maps. The works [70, 71] and [77] use the softmax function without scaling to implement the ASC, while the methods in [64, 67] use it with a scaling.

The ASC constraint can also be implemented through regularization by adding a penalty term to the loss function that penalizes if the abundances do not sum-to-one. This can, however, increase the instability of the training process and the time needed for convergence. The techniques described in [70, 71] enforce the ASC through such regularization. A related implementation is to augment the data matrix and the decoder weight's matrix with constant vectors. The methods in [62, 68, 78, 82] use this to weakly enforce the ASC constraint.

2.7.7 Abundance regularizations

In order to further constrain the unmixing problem, reasonable assumptions or priors regarding the abundance fractions are used to construct regularization terms that force the latent codes or the learned representations to live in certain subspaces of the latent space which reduces its effective dimensionality. The most common assumption made is that the abundance maps are sparse. Increased sparsity can be achieved by ℓ_1-norm regularization, or more generally by ℓ_p-norm regularization, i.e., by an objective function of the form

$$\mathcal{L}_{\ell_p \text{ regularized}}(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k) = \mathcal{L}(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k) + \lambda \|\boldsymbol{s}_k\|_p^p, \tag{2.20}$$

where p is a real number between 0 and 1, and $\boldsymbol{s}_k$ is the abundance vector for spectrum x_k. The methods in [70, 71, 77, 84] all use ℓ_1-norm regularization to promote sparsity

of the abundances.

In [70], it is argued that different abundance maps are close to orthogonal since no more than two endmembers are usually mixed within one pixel. This work introduces a new prior based on this, named orthogonal sparse prior (OSP), given by

$$\mathcal{L}_{\text{OSP}}(\mathcal{B}) = \sum_{i<j} \frac{\mathcal{B}_{\cdot i} \cdot \mathcal{B}_{\cdot j}}{\|\mathcal{B}_{\cdot i}\|_2 \|\mathcal{B}_{\cdot j}\|_2}, \tag{2.21}$$

where $\mathcal{B} \in \mathbb{R}^{b \times R}$ is a batch of size b of generated abundance maps. An ablation study shows that the OSP regularization leads to more sparse abundance maps than the conventional norm based prior. In [83], an variational autoencoder is used as an abundance regularization to enforce the ASC constraint.

2.7.8 Weights and endmember regularizations

Weight decay, ℓ_2-norm regularization on the weights of the encoder and decoder, is commonly employed to reduce overfitting and for better generalization. It can be applieds by adding a term in the loss function as

$$\mathcal{L}_{\ell_2 \text{ regularized}}(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k) = \mathcal{L}(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k) + \lambda \|\boldsymbol{W}\|_2^2, \tag{2.22}$$

where

$$\|\boldsymbol{W}\|_2 = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} W_{ij}^2 \right)^{1/2},$$

and $\boldsymbol{W}$ is the weight matrix of the layer we wish to regularize. ℓ_2 regularization penalizes the sum of squares of the weights and thus encourages smaller weights. Unlike ℓ_1 regularization, it does not lead to sparse solutions. The works in [77,84] use ℓ_2 for the weights of both the encoder and the decoder. The method in [78] applies ℓ_{21} regularization to the encoder's weights with ℓ_{21}-norm being a rotational invariant ℓ_1-norm (R1-norm) defined as [85]

$$\|\boldsymbol{W}\|_{21} = \sum_{i=1}^{m} \left(\sum_{j=1}^{n} W_{ij}^2 \right)^{1/2},$$

for an $m \times n$ matrix $\boldsymbol{W}$.

Another well-known regularization of the endmembers, i.e., the weights of the decoder, is minimum volume regularization. This regularization aims to minimize the volume of the simplex defined by the origin and the endmembers, i.e., the columns of $\boldsymbol{W}_D$. All reconstructed spectra are guaranteed to lie inside this simplex, being convex

combinations of the endmembers. The volume of this simplex is often denoted by

$$\text{MinVol}(\boldsymbol{W}_D) = \|\det \boldsymbol{W}_D\|,$$

and so the loss function with minimum volume regularization has the form

$$\mathcal{L}_{\text{MV regularized}}(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k) = \mathcal{L}(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k) + \lambda \text{MinVol}(\boldsymbol{W}_D), \tag{2.23}$$

The methods [82, 83] employ this regularization.

2.7.9 Multitask learning

Recently, there have been two works for HU [67,68] that have utilized multitask learning (MTL). In the context of neural networks, MTL means that the network learns multiple related tasks in parallel and parameters are shared between the tasks, usually via shared layers [86]. Thus, the different tasks are implemented as different branches of one network, which can have multiple inputs and outputs, depending on the problem being solved [87]. The benefits of MTL in spectral unmixing autoencoders include the following:

- **Faster learning**. When tasks are correlated, they will contribute to the aggregate gradient during backpropagation and increase the effective learning rate on the input to hidden layer weights [86]. This means that useful features will form faster in the shared hidden layer of the network.

- **Reduced risk of overfitting**. It has been shown that when layers are shared between tasks, the risk of overfitting the shared parameters can be up to an order N smaller, where N is the number of tasks [88].

- **Improved stability.** It has been shown that MTL tasks prefer hidden layer representations that other tasks prefer [86]. For MTL autoencoders where each task is unmixing a different pixel of a patch, and can lead to an increased stability and consistency of the method.

- **Incorporation of spatial information**. MTL allows for the construction of autoencoders that can unmix a whole patch at a time by having one task for unmixing each pixel from the patch. This allows for fully connected networks that can directly exploit the spatial correlation in the HSI with all the above-listed benefits of MTL.

In [67], MTL is utilized for spectral-spatial unmixing, and in [68], MTL is utilized to perform bilinear mixing model (BMM) spectral unmixing. One task performs the linear unmixing and another task does updates the bilinear components.

2.7.10 Approaches utilizing adversarial and variational autoencoders

Recently, there have been several works published in HU that utilize variational autoencoders. In [83], a VAE is used to ensure the nonnegativity and sum-to-one constraints of the abundances. The work in [89], employs a VAE to learn a spectral variability model and generate endmembers. They are then used in the solution of a spectral unmixing problem, cast as an alternating nonlinear least-squares problem that is then solved iteratively. Alternately adjusting the abundances and the low-dimensional representations of the endmembers in the generative model.

Recently, several works where adversarial autoencoders and generative models are used for unmixing have been published. The work in [90] proposes a joint metric neural network in the form of an adversarial autoencoder, where the Wasserstein distance between features in the discriminator is used to regularize the autoencoder, which is using SAD as the loss function. The Wasserstein distance between features of real and reconstructed spectra provide useful gradient information that promotes the autoencoder to reach a solution with better unmixing performance.

In [91], an abundance estimation method is proposed using 1D convolution kernels and spectral uncertainty. High-level representations are computed and are further modelled with the Multinomial Mixture Model (MMM) to estimate abundance fractions under high spectral uncertainty. A new trainable uncertainty term based on a nonlinear neural network is used in the reconstruction step. The uncertainty models are optimized by Wasserstein generative adversarial network to improve stability and capture uncertainty.

In [92], a method which uses a general perceptual loss-constrained adversarial autoencoder network to improve the unmixing performance is proposed. The encoder compresses the input pixels into hidden representations, which are estimates of the abundances of the materials while the decoder reconstructs the input pixels using the estimated abundances and endmembers. The network is trained to minimize the reconstructed error between the input and reconstructed pixels. A discriminator network is used to distinguish between the real and reconstructed pixels and is trained to maximize the discrepancy between the two. The method combines a general perceptual loss with the adversarial loss to further improve the consistency of high-level representations. The perceptual loss is calculated using hidden layers of the discriminator network to capture the feature difference between the input and reconstructed pixels.

2.7.11 Utilization of spatial information

Like other natural images, hyperspectral images have highly correlated pixels, and it is very desirable to make use of this spatial information. However, most unmixing methods are strictly spectral and do not directly exploit the spatial structure of hy-

perspectral images. This means that the methods work with one pixel at a time, even though it may use regularizations based on assumptions about the spatial structure, which will not be considered here to count as directly exploiting the spatial structure. At the time of writing, there are quite a few approaches based on autoencoders that directly exploit the spatial structure of HSIs. The earliest methods in this category are the method in [67] directly unmixes a patch at a time using a multiple branch architecture that is inspired by multitask learning (MTL) [86, 87].

The technique proposed in [64] was the first fully convolutional autoencoder for spectral unmixing and is inherently spatial in nature. It exploits the spatial and spectral structure of HSIs both for endmember and abundance map estimation. By working directly with patches of HSIs and not using pooling or upsampling layers, it preserves the spatial structure throughout the network and abundance maps arise naturally as feature maps of a hidden convolutional layer. This makes it very easy to apply complex spatial regularization on the abundances such as total variation (TV). However, the technique utilized 2D convolutions instead of 3D, which is not optimal as it only introduces a spectral bottleneck, similar to a non-convolutional autoencoder, instead of utilizing convolutions along the spectral dimension to better extract spectral features.

In [93], the authors introduce a framework that utilizes 3-D convolutional neural networks to jointly learn spectral-spatial priors and integrate spatial information into unmixing methods. Additionally, the framework includes a carefully designed decoder to handle endmember variability and employs a variational inference strategy to introduce uncertainty to endmembers. To prevent overfitting, the encoder networks are subjected to structured sparsity regularizers, and an $\ell_{2,1}$ loss is incorporated to ensure sparseness in the estimated abundances.

Another approach to utilize and integrate spatial information into the unmixing process is the use of multi- or two-stream networks, where two or more autoencoder networks work in collaboration. There have been a few works published that utilize this strategy. The study in [94] introduces the SSCU-Net, a novel spatial-spectral collaborative unmixing network that comprises a spatial AE network and a spectral AE network. The design of this network is rooted in the understanding that spatial information more effectively informs endmember extraction while spectral information better aids abundance estimation in AE-based unmixing networks. To leverage this discovery, the SSCU-Net employs a two-stream deep network architecture, incorporating a collaborative strategy that focuses on both abundance fractions and endmember information, ensuring efficient end-to-end training. Additionally, the network integrates a new superpixel utilization method by incorporating superpixels into the AE network, enhancing the use of spatial information. On the spectral side, the AE network uses spectral convolution to maximize spectral information utilization.

In [95], a novel approach for deep spatial-spectral feature extraction in HSI is introduced using a spatial-spectral feature learning network named ACAE. This method employs Dual-branch CNN and LCA-ResBlocks to optimize spatial-spectral feature extraction. The method innovatively employs adaptive convolution and local context

adaptive residual blocks to harness spatial details from hyperspectral images. Additionally, sparse constraints based on pixel correlations are applied to the loss function to bolster spatial information utilization. The dual-branch encoding network distinctively separates and then fuses spectral and spatial features, utilizing a convolutional network to further enhance their integration.

2.7.12 Endmember number estimation and robustness to dead pixels and noise

As was mentioned in Section 1.1.3, the first step in hyperspectral unmixing of real world images is estimating the number of endmembers in the scene. However, all of the methods discussed here do not do this and require that the number of endmembers is known beforehand. This is partly because the number of endmembers is an important architectural parameter that determines the number of hidden units in unmixing autoencoders and also because the endmember number estimation can be done separately. However, there are methods, such as the one in [96] that incorporate the estimation of the number of endmembers into the method pipeline. The method is a nonlinear unmixing method that has some other innovative features worth discussing here. One is the use of a the radial basis function (RBF) for application of the kernel trick [97] in a custom layer in the encoder to determine the abundances. The endmembers are estimated to be the centers of the RBFs, estimated using K-means clustering in the kernelized space, and the abundances are estimated as the normalized distances to the centers in the kernelized space. Kernel methods can be effective in unmixing as they can map the data to a higher dimensional space where the data is is more linearly separable [97]. Another feature is a spatial averaging filter that is applied to every pixel. The filter is a $n \times n$ averaging filter where the weights are determined by the RBF distances of neihbouring pixels from the center pixel. This filtering can greatly increase the robustness of the method to dead pixels and noise as the adjacent pixels are likely to contain information from the same material. Most of the methods discussed here do not take any specal measures to increase robustness to corrupt data as result of dead pixels or other equipment malfunction. Autoencoder methods are still fairly robust to noise in general as they inherently denoise the data as a result of the compression and reconstruction process. Another way to deal with dead pixels and similar corruption would be to selectively filter them out before training.

2.7.13 Hyperparameter Selection

Tuning hyperparameters is a big issue with deep learning-based methods and lot of effort has been devoted to the problem. If $\mathcal{A}$ is a machine learning algorithm with N hyperparameters having domains, $\Lambda_1, \cdots, \Lambda_N$, the hyperparmeter configuration space is N dimensional, $\mathbf{\Lambda} = \Lambda_1 \times \Lambda_2 \times \cdots \Lambda_N$. A particular hyperparameter configuration is then a vector, $\boldsymbol{\lambda}$, in this space and the algorithm instantiated with these parameters can be denoted $\mathcal{A}_{\boldsymbol{\lambda}}$. The general problem of hyperparameter tuning can then be stated

as [98]

$$\boldsymbol{\lambda}^* = \arg\min_{\boldsymbol{\lambda}\in\mathcal{A}} \mathbb{E}_{(D_{\text{train}},D_{\text{valid}})\sim\mathcal{D}} \boldsymbol{V}(\mathcal{L},\mathcal{A}_{\boldsymbol{\lambda}},\boldsymbol{D}_{\text{train}},\boldsymbol{D}_{\text{valid}}). \tag{2.24}$$

Here, $\boldsymbol{V}(\mathcal{L},\mathcal{A}_{\boldsymbol{\lambda}},\boldsymbol{D}_{\text{train}},\boldsymbol{D}_{\text{valid}})$, measures the loss of the model $\mathcal{A}$ with hyperparameters $\boldsymbol{\lambda}$ trained on $\boldsymbol{D}_{\text{train}}$ and evaluated on validation data $\boldsymbol{D}_{\text{valid}}$.

In deep learning, algebraic models are usually unavailable, and the model usually has to be treated as a blackbox when tuning hyperparameters. Generally, any blackbox optimization method can be applied to hyperparameter tuning of deep learning models. Two standard but primitive methods are grid search and random search. In grid search, a finite set of values of each hyperparameter are specified, and the model evaluated on the Cartesian product of these sets. Grid search suffers from the curse of dimensionality as the required number of evaluations grows exponentially with the dimension of the configuration space [98]. Random search, which samples configurations randomly, works usually better than grid search when some hyperparameters are much more important than others [99].

In recent years, Bayesian optimization [100] has emerged as a state-of-the-art optimization framework for the optimization of blackbox functions, especially for autoencoder methods. The main reason for this is that the network's actual loss is not a measure of its unmixing performance. Instead, the network's loss is a similarity measure between the input and the reconstructed input. However, the unmixing performance measures the quality of the weights of the linear decoder (the endmembers) and the abundances that are the encoder part's latent codes. Thus, it cannot be said that autoencoder methods converge to a solution regarding the endmembers or abundances, in contrast with many traditional methods.

It is, therefore, challenging to use traditional hyperparameter tuning methods such as Bayesian hyperparameter optimization [100] or maximum-a-posteriori (MAP) [101] to select hyperparameters for autoencoder unmixing methods. This is in part because the number of epochs used for training is one of the hyperparameters. Also, because of random weight initialization, many runs are required per hyperparameter configuration to get a reliable measure for any automatic or manual hyperparameter selection. Doing hyperparameter grid search is often the only way to reliably select hyperparameters, but it can be very time-consuming as it requires a significant number of runs. For the experiments in this article, the hyperparameter values used for the methods were the ones recommended by their authors.

2.7.14 Why do autoencoders work so well compared to traditional methods?

Why do autoencoders work well for spectral unmixing compared to traditional methods? What exactly is it that makes them perform so well? A linear nonnegative autoencoder, having single layer encoders and decoders and using the MSE loss function, essentially performs NMF and should perform similarly to NMF methods in

unmixing. However, autoencoders allow any loss functions such as SAD, whereas traditional methods use the MSE. Additionally, when using more complex architectures, the following can be listed as advantages of autoencoders over traditional methods:

1. **Hierarchical feature learning**: Deep autoencoders, having multiple layers, are able to learn a hierarchy of features from the data. The initial layers often capture low-level patterns, while the deeper layers represent more abstract and composite features. This hierarchical representation allows them to better capture the intricacies of spectral data.

2. **Data compression and noise reduction**: Autoencoders are trained to compress data into a lower-dimensional latent space and then reconstruct it. This can inherently denoise the data to some extent, focusing on the most salient features, which can be beneficial in cases where the spectral data is noisy.

3. **Modeling of non-linearities**: Autoencoders, being neural networks, can model non-linear relationships within the data. This is particularly useful in cases where spectral mixing is not purely linear, allowing autoencoders to capture more complex interactions between spectral components.

4. **Flexibility**: Autoencoders implemented with deep learning frameworks can be easily customized and integrated with other neural network architectures, such as convolutional layers for handling spatial information in hyperspectral images, which can enhance the spectral unmixing performance. It is easy to implement complex mixing models and regularizations.

5. **Incorporating Priors**: While traditional methods can incorporate certain priors, deep learning based autoencoder models can be more easily designed to integrate more complex prior information about the data or the unmixing process using specialized layers or loss functions.

6. **Scalability**: When implemented on GPUs, deep learning models can handle large datasets much more efficiently. This scalability allows them to be trained on larger amounts of data, potentially leading to better and more robust models.

The option to use scale-invariant loss functions such as SAD gives autoencoders a significant advantage over traditional methods for linear unmixing. Also, the encoder and the decoder can be arbitrarily nonlinear, which can further improve performance, especially abundances. In the experimental section an ablation study will be performed using a simple spectral unmixing autoencoder to try to answer this question.

2.8 Non-Blind Methods Based on Autoencoders

Until now, only blind unmixing methods based on autoencoders have been discussed. Non-blind approaches are methods that estimate the abundances of endmembers but

not the endmembers themselves. Many methods use neural networks for abundance estimation, and in most cases, such networks are essentially autoencoders or utilize autoencoders. This section will discuss various non-blind approaches that are based on autoencoders but it not meant to be comprehensive review of such methods.

An autoencoder is simply a neural network trained to reconstruct its input and is prevented from learning the identity function. We have seen a direct correspondence between the weights of a linear decoder in a spectral unmixing autoencoder and the endmembers matrix, according to the LMM. If we have a set of endmembers given, they can be used as the weights of a non-trainable linear decoder, i.e., they are prevented from changing during training which then only trains the encoder part. Such a network is still an autoencoder but works like a nonlinear regression method that determines the abundances maps corresponding to the given endmembers.

This approach is used in [102], where the encoder part of an spectral unmixing network is improved to utilize convolution layers to enable deeper architectures of the encoder and a special spectral normalization layers are used instead of batch normalization. The decoder has fixed weights with endmembers coming from another method. Another method based on this concept is [103] where the encoder is a deep convolutional network using a deep prior. The fixed decoder approach can be generalized to use a whole spectral library of endmembers, resulting in methods that are a combination of autoencoder unmixing and sparse regression based methods using a spectral library.

The technique in [104] is an example of such a method. The encoder has a deep convolutional architecture, but the last two layers are fully connected and use the softmax activation, essentially a classifier that picks the most suitable endmembers out of many. The input to the decoder layer has a much higher number of units than the number of endmembers to be estimated, but most of these activations will be zero since the output of the encoder is required to be sparse.

Another approach for abundance estimation utilizes multitask learning (multi-stream networks) and involves two different networks or tasks that share layers. A reconstructing autoencoder network is guided by another network that shares layers with the autoencoder in this approach. This guiding network is typically trained to learn the relationship between endmember candidates obtained by another endmember extraction method and their abundances as determined by the LMM. This network will guide the reconstructing autoencoder and make its encoder part encode spectra to abundances. The works [105, 106] are examples of this type of approach.

A variation of this is the method described in [107], where the guiding network is itself a spectral unmixing autoencoder that shares decoder weights with another spectral unmixing autoencoder. The guiding network is trained on endmember bundles obtained by VCA from superpixel segmentation of an HSI. Both networks are encouraged to produce latent codes that follow the Dirichlet distribution, i.e., sum-to-one.

This short overview of non-blind methods illustrates further how powerful and widespread

the autoencoder architecture is becoming in hyperspectral unmixing.

Chapter 3

Spatial-Spectral Hyperspectral Unmixing Using Multitask Learning

Hyperspectral images, like other natural images, have highly correlated pixels and it is very desirable to make use of this spatial information. In this chapter, a deep learning based method for blind hyperspectral unmixing is introduced. The method uses multitask learning through multiple parallel autoencoders to unmix a neighborhood of pixels simultaneously. Operating on image patches instead of single pixels enables the method to take advantage of spatial information in the hyperspectral image. The method is the first in its class to directly utilize the spatial structure of hyperspectral images (HSIs) for the estimation of the spectral signatures of endmembers in the data cube. We evaluate the proposed method using two real HSIs and compare it to seven state-of-the-art methods that either rely only on spectral or both on spectral and spatial information in the HSIs. The proposed method outperforms all the baseline unmixing methods in experiments

3.1 Introduction

As in most natural images, there is a strong spatial correlation between pixels in an HSI. The majority of HSU methods developed so far do not exploit the spatial structure of HSIs but methods that do so are called spectral-spatial methods. What such spectral -spatial methods have in common, is the use of natural assumptions about the spatial correlation of pixels in an HSI as priors to control sparsity and smoothness of the obtained abundance maps.

In this chapter, a novel autoencoder based method which attempts to make direct use of spatial information from neighboring pixels in an HSI is introduced. The method directly unmixes a whole neighborhood of pixels at a time using an architecture that is inspired by multitask learning (MTL) [86, 87]. It consists of multiple branches of unmixing autoencoder, each tasked with unmixing a single pixel in a neighborhood, that share features between them. Even though the tasks are identical, MTL is still beneficial. The main benefits of MTL are the following:

- **Faster learning**. Correlated tasks contribute to the aggregate gradient during backpropagation and thus increase the effective learning rate on the input to hidden layer weights [86]. Useful features form faster in the shared hidden layer of the network.

- **Reduced risk of overfitting**. Having shared hidden layers can greatly reduce the risk of overfitting. It has been shown that the risk of overfitting shared parameters can be up to an order N smaller, where N is the number of tasks, than overfitting task-specific parameters [88].
- **Improved stability.** For some HSI datasets with correlated and underrepresented endmembers, a nonnegative sparse autoencoder (NNSAE) often finds two (or more) very different solutions that correspond to different local minima, with one of them preferred. It can be difficult to ensure consistency in such cases, i.e., that the network always chooses the preferred solution. It has been demonstrated that MTL tasks prefer hidden layer representations that other tasks prefer [86]. In our case, this along with the fact that all autoencoders share a decoder, means that the preferred solution becomes even more preferred in an MTL setting, leading to increased stability.
- **Incorporation of spatial information**. Through the sharing of the first hidden layer between autoencoders, each autoencoder or task has access to all features from all the pixels input to the network. By selecting these pixels from a neighborhood in the HSI, we are exploiting the spatial correlation in the HSI, i.e., the assumption that all the pixels from a small neighborhood should have similar abundances. This reinforces the network's learning of shared features between pixels and enhances the effectiveness of the multitask learning.

In more traditional methods, i.e., non-neural network methods, MTL is often implemented by capturing the relationship of multiple related tasks using a low-rank structure through a nuclear norm regularization, and identifying the outlier tasks using a group-sparse structure [108].

To summarize, the main contributions of the proposed method and improvements over our previous method described in [66] are the following:

- The method makes direct use of the spatial information in an HSI by unmixing a whole patch at a time using multiple parallel autoencoders. Hence, the major difference is that the proposed method is a spectral-spatial unmixing method, while the previous method is a spectral unmixing method.
- The method uses the softmax function to enforce the abundance sum-to-one constraint.
- The method makes better use of batch normalization and dropout than our previous method.

The experimental section aims to confirm that the incorporation of spatial information using multiple tasks gives better results than a single task autoencoder. The proposed method is a substantial improvement over our previous autoencoder method [66] as can be seen in the experimental section. It significantly outperforms all the comparison methods, especially regarding consistency and has much lower variance.

3.1.1 Spectral-spatial methods

Spectral-spatial methods are unmixing methods that seek to exploit the spatial relationship in HSIs. Such methods most often introduce weighing factors to penalize non-zero coefficients on the sparse solution or regularization terms based on the abundance fractions to enhance sparsity and sharpness of abundance maps. A good example of such a method and one of the first methods to exploit the rich spatial structure of HSIs was the SUnSAL-TV method [109].

Another similar method is [110]. Other examples are methods using spectral–spatial weighted sparse regression described in [111]. In [44,112,113], a TV regularization term is used as a way of exploiting spatial homogeneity in the image while still retaining sharp edges. Examples of NMF methods using spatial regularization are [114,115]. An example of a hybrid linear and nonlinear NMF unmixing method based on a spatial prior is given in [116].

Another class of spatial-spectral methods are superpixel segmentation methods. These methods use a superpixel segmentation method as preprocessing technique to hyperspectral images. Following this, the unmixing is applied directly on the mean spectra of the various image segments. Such processing significantly improves the signal quality of the data and reduces the ill effects of noise. Examples of this are [117–119] and [120] which applies a spatial group sparsity regularization derived from segmentation. What all these methods have in common, is the use of natural assumptions about the spatial correlation of pixels in an HSI as priors to control sparsity and smoothness of the obtained abundance maps.

3.2 The method

Fig. 3.1 shows a schematic of the method. There are k^2 tasks in the form of autoencoders, one for each pixel in a $k \times k$ neighborhood $\mathcal{N}_k$, input to the method. The inputs are concatenated and connected into a large hidden layer shared by all the autoencoders. This shared hidden layer enables all the unmixing tasks to access features from all the neighboring pixels. Such sharing of representations between tasks can improve both learning and generalization [86], and is called hard parameter sharing in the language of MTL.

The architecture branches again after the shared layer. Each autoencoder performing further dimensional reduction through an additional hidden layer, called A_i in Fig. 3.1. The activations for each autoencoder become the abundance fractions after applying the abundance sum-to-one constraint (ASC). However, since endmembers are a property of an HSI as a whole, all autoencoders are required to use the same decoder having weights $\boldsymbol{W}_D$, placing a constraint on each of them to discover the same endmembers. This is what connects all the tasks. If one tasks modifies the endmembers, the endmembers are modified for all the tasks.

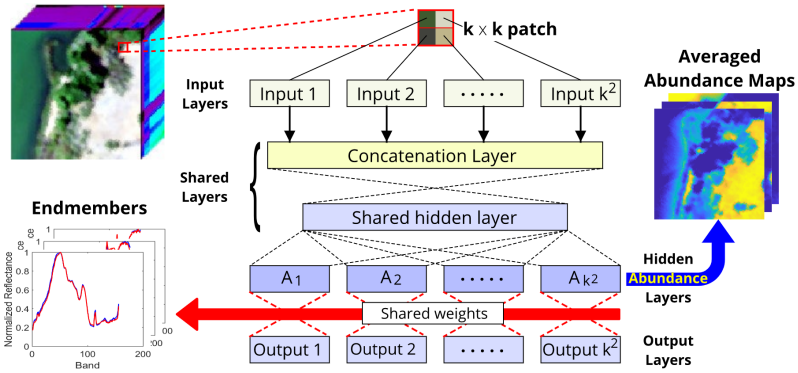


Figure 3.1: A schematic of the proposed method.

Table 3.1: The layers/transformations in each encoder and their expressions listed sequentially.

	Layer/Transform	Expression
1	Input i	$\boldsymbol{a}_i^{(0)} = \boldsymbol{x}_i^{\mathcal{N}_k^{(p)}}$
2	Concatenation	$\boldsymbol{a}^{(1)} = [\boldsymbol{a}_1^{(0)}, \cdots, \boldsymbol{a}_i^{(1)}, \cdots, \boldsymbol{a}_{k^2}^{(1)}] = \boldsymbol{C}\boldsymbol{a}_i^{(0)}$
3	Hidden layer shared	$\boldsymbol{a}^{(2)} = g(\boldsymbol{W}^{(2)}\boldsymbol{a}^{(1)})$
4	Batch normalization	$\boldsymbol{a}_i^{(3)} = \mathrm{BN}\,\boldsymbol{a}_i^{(2)}$
5	Hidden layer not shared	$\boldsymbol{a}_i^{(4)} = g(\boldsymbol{W}_i^{(4)}\boldsymbol{a}^{(3)})$
6	Batch Normalization	$\boldsymbol{a}_i^{(5)} = \mathrm{BN}\,\boldsymbol{a}_i^{(4)}$
7	ASC constraint	$a_{i,j}^{(7)} = \frac{e^{\alpha a_{i,j}^{(6)}}}{\sum_{j=1}^{R} e^{\alpha a_{i,j}^{(6)}}} = \boldsymbol{\sigma}_{\boldsymbol{\alpha}}\boldsymbol{a}_i^{(6)}$

Now, it might be asked whether using a single autoencoder with single augmented input, consisting of the $k \times k$ neighborhood vectorized, might give similar benefits. The problem with that approach is that the reconstruction loss could only consider the reconstrution of the center pixel in the neighborhood in order to get meaningful endmembers with the correct number of bands. The autoencoder would not be sufficiently forced to utilize the extra information in the large input since the reconstruction of the extra pixels is not considered.

3.2.1 Loss function

Each NNSAE consists of an encoder part, $\mathcal{G}_E$, and a decoder part, $\mathcal{G}_D$. The encoder encodes the input $\boldsymbol{x}_p$ into a hidden representation $\boldsymbol{h}_p = \mathcal{G}_E(\boldsymbol{x}_p)$. The encoder of autoencoder i consists of the layers or transformations listed sequentially in Table 3.1. We have used operator notation to denote the operations given in rows 2, 4, 6, and 7. From the table we can construct an expression for the hidden representation $\boldsymbol{h}_p$, obtained by a feed-forward pass, as

$$\mathcal{G}_E(\boldsymbol{x}_i^{\mathcal{N}_k}) = \boldsymbol{\sigma}_{\boldsymbol{\alpha}} \,\mathrm{BN}\, g(\boldsymbol{W}_i^{(3)} \mathrm{BN}\, g(\boldsymbol{W}^{(2)} \boldsymbol{C} \boldsymbol{a}_i^{(0)})). \tag{3.1}$$

Here g is the activation function which is the LeakyReLU [53] function, and $\boldsymbol{\sigma}_{\boldsymbol{\alpha}}$ is the softmax function with scaling parameter α which enforces the ASC constraint according to row 7 in Table 3.1. We apply dropout [79] after the shared hidden layer using a dropout rate of 0.5. We also apply batch normalization [72], denoted BN, after both hidden layers. The decoder then reconstructs the input as $\hat{\boldsymbol{x}}_p$ from the hidden representation as

$$\hat{\boldsymbol{x}}_p = \mathcal{G}_D(\boldsymbol{h}_p). \tag{3.2}$$

The final layer in the network is the decoder having weights $\boldsymbol{W}_D$ which are constrained to be nonnegative, and which has linear activation as required by the LMM. The action of the decoder is a simple linear transformation which can be written as

$$\mathcal{G}_D(\boldsymbol{h}_i^{N_k^{(p)}}) = \boldsymbol{W}_D \boldsymbol{h}_i^{N_k^{(p)}}, \tag{3.3}$$

where $\boldsymbol{h}_i^{N_k^{(p)}}$ is the hidden representation belonging to pixel i in the neighborhood $N_k^{(p)}$ which is the input to the method. The loss function we use for each autoencoder i is the spectral angle distance (SAD) and is given by

$$\mathcal{L}^{(i)} = \cos^{-1}\left(\frac{\langle \boldsymbol{x}_i, \hat{\boldsymbol{x}}_i \rangle}{\|\boldsymbol{x}_i\|_2 \, \|\hat{\boldsymbol{x}}_i\|_2}\right).$$

The total loss of the neural network is the sum of each individual autoencoder loss. The method divides the HSI into a collection of neighborhoods $\mathcal{N}_k$ and unmixes one neighborhood at a time. The neighborhood loss is the sum of the individual pixel

(A) Samson (B) Urban

Figure 3.2: Simulated RGB images of the HSIs used in experiments.

losses for each pixel in the neighborhood

$$\mathcal{L}^{\mathcal{N}_k(\boldsymbol{x}_p)} = \sum_{i=1}^{k^2} \mathcal{L}^{(i)}, \tag{3.4}$$

and the total loss is the sum of the neighborhood losses for all $k \times k$ neighborhoods in the training dataset consisting of N randomly chosen patches from the HSI

$$\mathcal{L} = \sum_{p=1}^{N} \mathcal{L}^{\mathcal{N}_k(\boldsymbol{x}_p)}. \tag{3.5}$$

At the end of training, the decoder weights matrix, $\boldsymbol{W}_D$, contains the endmembers and the abundances of each pixel $\boldsymbol{x}_p$ are the activations $\boldsymbol{h}_p$ in the last hidden layer before the decoder. These can be extracted by doing prediction for the whole HSI using a network consisting only of the already trained encoder.

3.3 Experiments

Experiments were conducted on the Urban and Samson datasets. We quantitatively evaluate the endmembers extracted for each dataset using 1.6 and the extracted abundance maps using 1.7.

We evaluate the proposed multitask autoencoder unxmixing (MTAEU) method using $k \times k$ neighborhoods for $k = 2, 3, 4, 5$. Also included in the experiments are the results of a single task autoencoder (STAEU) having the same architecture as the MTAEU network. We compare our results for extracted endmembers and abundance maps to seven other methods, six of which perform blind unmixing, one method that only extracts endmembers. Three of the comparison methods are based on deep learning. The comparison methods are vertex component analysis (VCA) [36], sparsity constrained nonnegative matrix factorization ($\ell_{1/2}$-NMF) [39], sticky hierarchical Dirichlet process

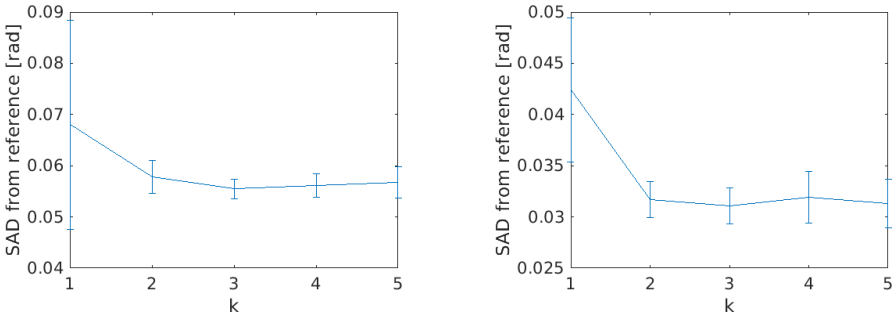


Figure 3.3: Mean SAD from reference for both datasets versus the width of patch, k. The plot for the Urban (4 endmembers) dataset is on the left and the plot for the Samson dataset is on the right.

(SHDP) [121] which is a spatial-spectral blind unmixing method, spatial group sparsity regularized nonnegative matrix factorization (SGSRNMF) which is a spatial-spectral blind unmixing method [120], deep autoencoder unmixing (DAEU), an autoencoder based method described in [66], stacked nonnegative sparse autoencoder (SNSA) unmixing method described in [82], and an untied denoising autoencoder With sparsity (uDAS) unmixing method described in [62].

All autoencoder methods except for SNSA and uDAS are initialized randomly, while the rest of the methods are all initialized or partially initialized using VCA. The activation function used for the autoencoder methods is the LeakyReLU function and the number of randomly selected patches used in the training for the proposed method was $N = 2000$ for the Urban dataset and $N = 300$ for the Samson dataset. The optimizer used was the RMSProp [122] optimizer having learning rate 0.02 and learning rate decay of 0.02. The networks were trained for 100 epochs.

Fig. 3.3 shows the average SAD from 25 runs in radians from the reference for both datasets as a function of k for $k = 1, \cdots, 5$. The error bars are the standard deviation. It can be seen from the figure that MTL is beneficial in both cases, especially for the Urban dataset and also that $k = 3$ achieves the lowest SAD score for both datasets and also the least variance.

Fig. 3.4 shows the result of an experiment designed to evaluate the effect of selecting the pixels in a patch in their original spatial arrangement versus selecting the input pixels randomly. Such an experiment can confirm whether the method is using the spatial information in the scene. The experiment was performed using the Urban dataset for four different sizes of $k \times k$ neighborhoods. Fig. 3.4 shows a boxplot of the results of 25 runs for each value of k. The figure shows clearly that keeping the spatial arrangement of the pixels gives much better average SAD from the reference and less variance than selecting the pixels randomly. The results are in line with the benefits of MTL discussed earlier. It must be noted that if the number of neighborhoods used in training is increased, the difference becomes somewhat less, which shows also that

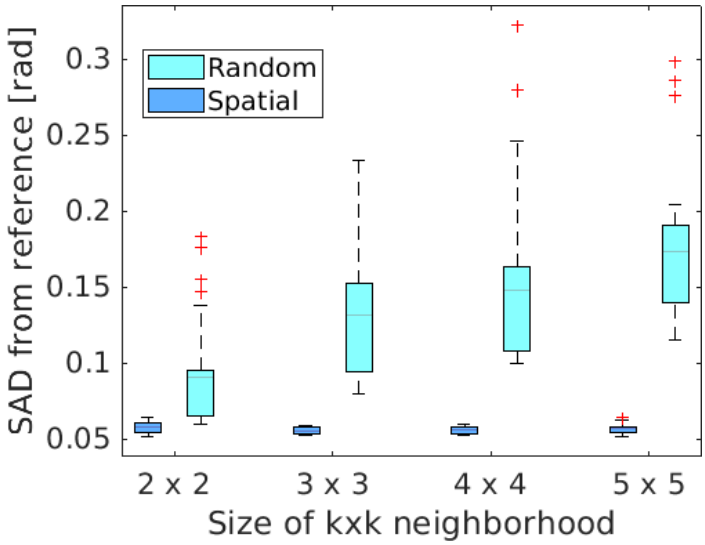


Figure 3.4: Boxplot of SAD from ground truth for the Urban dataset (4 endmembers) showing the effect of selecting pixels into patches randomly vs. spatially for three different patch sizes. Number of neighborhoods used in training is 2000 and number of epochs is 100.

spatial arrangement speeds up the convergence of the method greatly. Another trend which the figure depicts well, is the more spatial the method becomes, i.e., as more neighboring pixels are used, the greater the difference between spatial and random arrangement becomes for a fixed number of neighborhoods used in training.

Table 3.2 shows the SAD in radians of extracted endmembers from the reference, both for individual endmembers and the average, for the proposed method and the comparison methods for the Urban dataset using four reference endmembers. Fig. 3.5 shows the endmembers extracted by all the methods except VCA. The blue curves are the extracted endmembers (from 25 runs) and the red curves are the reference endmembers. When calculating SAD scores and displaying endmembers in plots, the endmember solutions are matched to the most similar reference endmembers as measured by the SAD metric. This is needed because the order of endmembers returned by the autoencoder methods is essentially random between runs.

It can be seen from both Table 3.2 and Fig. 3.5 that the autoencoder methods are in a class of their own compared to the other methods used in the experiments. The MTAEU method achieves the best score for all endmembers except the Tree endmember and also the lowest average SAD score, whereas STAEU achieves the lowest SAD score for the Tree endmember. This should not be over-interpreted, the scores which are averages, are similar and also the reference is should not be considered some final truth. What is more important, is the fact that the proposed method has a lower variance. A consistent method having low variance is preferable over method which

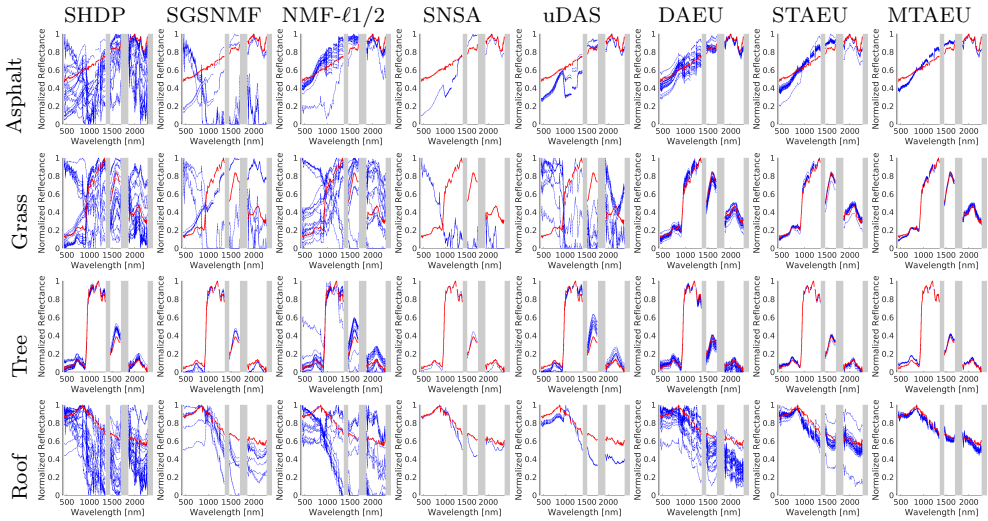


Figure 3.5: The endmember spectra extracted by the proposed method using 3x3 pixel neighborhood and the comparison methods for the Urban data set with 4 endmembers. The red curves are the reference endmembers and the blue curves are extracted endmembers.

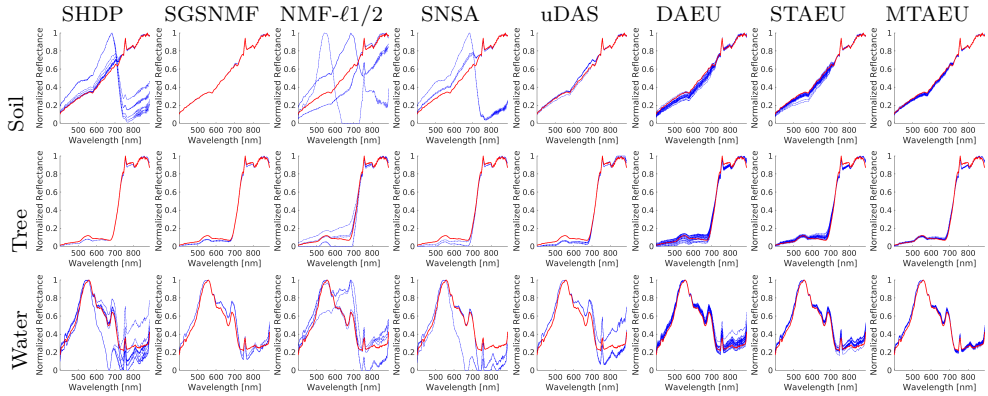


Figure 3.6: The endmember spectra extracted by the proposed method using 3x3 pixel neighborhood and the comparison methods for the Samson data set. The red curves are the reference endmembers and the blue curves are extracted endmembers.

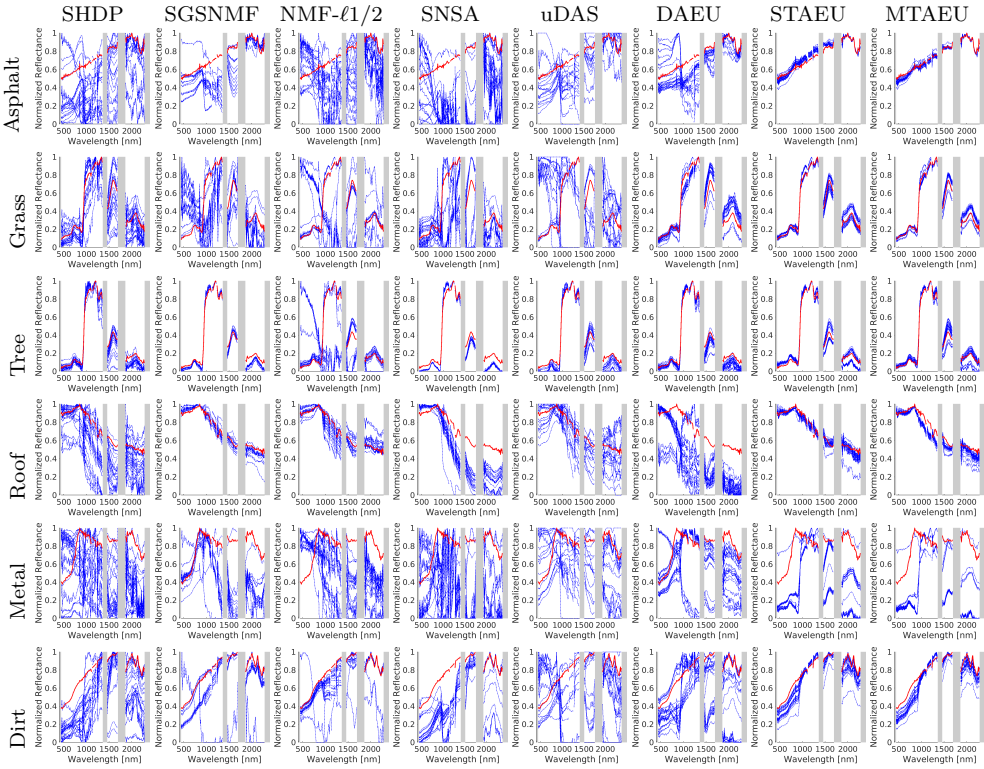


Figure 3.7: The endmember spectra extracted by the proposed method using 3x3 pixel neighborhood and the comparison methods for the Urban data set with 6 endmembers. The red curves are the reference endmembers and the blue curves are extracted endmembers.

Table 3.2: SAD($\times 10^{-2}$) in radians from reference for the Urban dataset with 4 reference endmember. The figures are the mean of 25 runs along with the standard deviation. Best results are in bold typeface.

Method/Endm	1 (Asphalt)	2 (Grass)	3 (Tree)	4 (Soil)	Average
VCA	21.9(3.1)	40.1(4.2)	20.2(8.6)	82.4(0.2)	41.2(1.9)
SHDP	33(14)	42(27)	8.8(4.1)	41(16)	31.1(3.9)
SGSRNMF	48(35)	63(28)	9.6(0.7)	21(11)	35.5(5.8)
NMF-ℓ1/2	15(10)	54(39)	15.7(9.4)	49(21)	29.6(5.9)
SNSA	29.12(0.04)	121.65(0.42)	7.77(0.01)	9.23(0.05)	41.9(0.1)
uDAS	18.1(3.3)	116(16)	15.0(3.9)	17.00(0.12)	41.5(4.1)
DAEU	7.2(2.4)	7.7(2.9)	7.1(3.0)	22(21)	10.9(2.4)
STAEU 1px	9.8(2.8)	4.6(1.5)	**3.7(0.8)**	9.3(5.4)	6.8(2.0)
MTAEU 9px	**8.4(0.5)**	**4.21(0.37)**	5.4(0.4)	**4.15(0.46)**	**5.6(0.2)**

Table 3.3: SAD($\times 10^{-2}$) in radians from reference for the Urban dataset with 6 reference endmembers. The figures are the mean of 25 runs along with the standard deviation. Best results are in bold typeface.

Method/Endm.	1 (Asphalt)	2 (Grass)	3 (Tree)	4 (Roof)	5 (Metal)	6 (Dirt)	Average
VCA	32(29)	55(42)	30.4(6.7)	28(11)	68(17)	106(32)	53.4(6.7)
SHDP	44(20)	29(12)	14.6(7.7)	30(17)	64(22)	39(18)	36.8(3.6)
SGSNMF	28.4(6.9)	55(17)	9.6(1.4)	5.8(3.1)	54(12)	18(20)	28.4(5.2)
NMF-ℓ1/2	44(21)	73(58)	45(56)	8.6(4.9)	59(23)	13(14)	41(3)
SNSA	72(25)	64(9)	17(1)	36.1(8.7)	81(20)	41.6(12.5)	52.0(2.4)
uDAS	29(14)	89(45)	14.4(5.3)	21.2(8.9)	43(22)	55(41)	41.7(4.9)
DAEU	19(11)	19.1(3.9)	13.0(4.7)	50.6(6.2)	**34(29)**	40(27)	29.3(3.5)
STAEU 1 px	5.0(1.2)	**8.5(2.2)**	10.3(2.9)	**6.2(2.8)**	56(21)	**9.7(5.2)**	**16.0(3.5)**
MTAEU 9 px	**3.87(0.83)**	9.0(2.5)	**8.3(4.5)**	7.5(2.1)	72(18)	10.9(3.8)	18.5(2.3)

Table 3.4: SAD($\times 10^{-2}$) in radians from reference for the Samson dataset. The figures are the mean of 25 runs along with the standard deviation. Best results are in bold typeface.

Method/Endm.	1 (Soil)	2 (Tree)	3 (Water)	Average
VCA	12(24)	4.9(0.3)	13.0(0.1)	9.7(7.3)
NMF-ℓ1/2	5(18)	3.6(0.6)	4.3(0.2)	4.2(6.2)
SHDP	2(34)	3.70(0.04)	20.6(9.4)	15(14)
SGSRNMF	**0.86(0.01)**	3.95(0.20)	9.2(0.3)	4.68(0.03)
SNSA	0.25(32)	7.5(1.5)	28.4(6.9)	20(12)
uDAS	3.12(0.15)	5.5(0.5)	14.1(1.2)	7.6(0.5)
DAEU	2.7(4.6)	**3.3(0.3)**	3.9(0.4)	3.3(1.6)
STAEU 1 px	2.1(0.5)	5.6(1.0)	3.6(0.7)	3.7(0.4)
MTAEU 9 px	2.3(0.6)	3.7(0.3)	**3.38(0.31)**	**3.11(0.18)**

might achieve a slightly better SAD score w.r.t. some reference endmembers but has high variance. Also, the individual endmember scores are not entirely independent but are linked through the ASC constraint. A bad solution for one endmember can affect the solutions for the other endmembers if its abundance is significant. In the end, the MTAEU method does achieve better MSE score for the Tree abundance map than STAEU.

Fig. 3.5 shows very clearly that the MTAEU method has both a far less variance than the STAEU method and has far better consistency, i.e., no outlier solutions. In contrast, the comparison methods, aside from STAEU and SNSA, have bad consistency and large variance, especially the SHDP and NMF-$\ell 1/2$ methods. The low variance and the excellent consistency of the MTAEU method is directly in accordance with the benefits of MTL listed above in the introduction.

The extracted endmembers by MTAEU need to simultaneously give optimal reconstruction for k^2 autoencoders in parallel. As discussed earlier, multiple tasks increase stability. The endmembers are the weights of the last layer in the network which acts as a decoder for all the different encoder branches and are influenced by them all. Even if one branch seeks a solution in a different local minima, it will not have too much effect on the extracted endmembers of the whole network. The consistency of the method can thus be expected to increase with increasing k. This increased consistency is a clear benefit over ordinary single task deep learning based methods.

Table 3.3 shows the SAD in radians of extracted endmembers from the reference, both for individual endmembers and the average, for the proposed method and the comparison methods for the Urban dataset using six reference endmembers. Fig. 3.7 shows the endmembers extracted by all the methods except VCA. For six endmembers the table shows that the STAEU method achieves the lowest average SAD score for the Urban dataset. Fig. 3.5 shows the reason for this as it can be seen that both STAEU and MTAEU have trouble estimating the "Metal" endmember. In the Urban dataset, the Metal endmember is underrepresented and we have seen that in such situations the autoencoders can achieve lower global reconstruction error or loss by estimating the underrepresented endmember as a variant of a heavily represented endmember. A better estimate could be obtained by the autoencoder methods by increasing the number of patches used in training and/or applying regularization on abundances, but there is a trade-off between this number and the general quality of estimated endmembers.

As can be seen in Fig. 3.7, both the STAEU and MTAEU methods tend to estimate it as a variant of either the Tree or the Grass endmember. The SAD between the reference Grass endmember and the reference Metal endmember is lower than it is between the Tree endmember and Metal. For STAEU, the Metal endmember is more often similar to the reference Grass endmember than it is to the reference Tree endmember, giving a better average score for the Metal endmember. MTAEU, however, is more consistent in estimating the Metal endmember similar to the Tree one.

Table **??** shows the SAD from the reference for the Samson dataset. As before, the values in the table are the average of 25 experiments and the results shown for the proposed method are for the optimal neighborhood size, which was for 3×3 neighborhoods. The best results are in bold typeface. The MTAEU method achieves the best average SAD score on this dataset and the best SAD score for the Water endmember. More significantly, MTAEU has substantially lower variance than the single pixel method, STAEU. The DAEU method achieves the second best average score followed by the SGSRNMF method which has the least variance of all the methods. Fig. 3.6 shows the extracted endmembers and again the same pattern can be observed with MTAEU having considerably less variance and more consistency than STAEU.

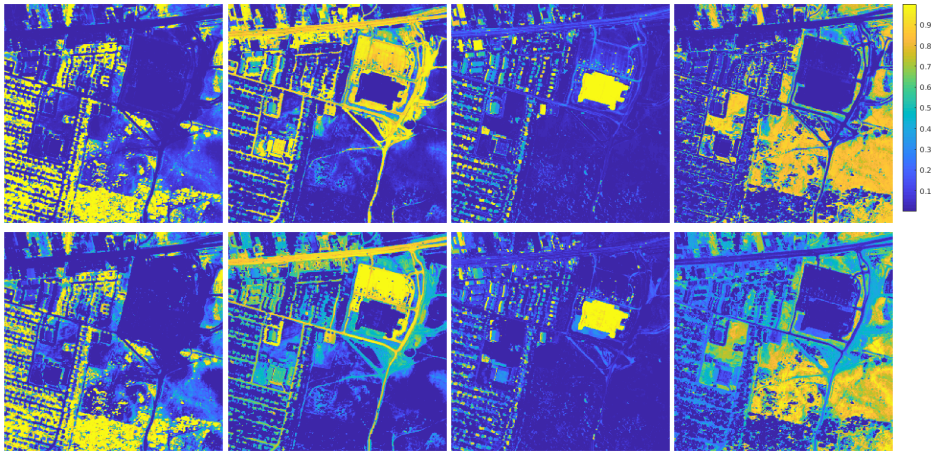


Figure 3.8: The abundance maps produced by MTAEU for the Urban dataset (top row) and the reference abundance maps (bottom row). The first column is the abundance for the "Tree" endmember, the second column the "Asphalt" endmember, the third column is the "Roof" endmember, and the last column the "Grass" endmember.

Table 3.5: MSE ($\times 10^{-2}$) between extracted abundance maps and the reference abundance maps for all methods for the Urban dataset with four reference endmembers. The figures are the mean of 25 runs along with the standard deviation. Best results are in bold typeface.

Methods	1 (Asphalt)	2 (Grass)	3 (Tree)	4 (Soil)	Average
SHDP	9.5(2.0)	13.5(7.1)	8.3(2.4)	7.0(5.5)	9.6(2.0)
SGSNMF	14.0(2.5)	13.0(4.3)	7.2(1.8)	3.6(1.8)	9.4(1.0)
NMF-ℓ1/2	17.2(0.3)	20.5(0.4)	18.9(0.3)	5.72(0.06)	15.6(0.1)
SNSA	11.3(0.1)	16.80(0.05)	12.60(0.07)	4.33(0.02)	11.26(0.02)
uDAS	10.1(0.4)	18.7(3.9)	9.2(1.1)	3.85(0.12)	10.5(0.7)
DAEU	3.1(1.8)	5.6(2.6)	3.9(3.4)	1.9(1.3)	3.6(1.5)
STAEU 1 px	2.7(0.8)	2.8(0.4)	0.9(0.3)	1.2(1.1)	1.9(0.6)
MTAEU 9 px	**2.3(0.1)**	**2.3(0.2)**	**0.7(0.1)**	**0.79(0.08)**	**1.50(0.08)**

Table 3.6: MSE ($\times 10^{-2}$) between extracted abundance maps and the reference abundance maps for all methods for the Urban dataset with six reference endmembers. The figures are the mean of 25 runs along with the standard deviation. Best results are in bold typeface.

Methods	1 (Asphalt)	2 (Grass)	3 (Tree)	4 (Roof)	5 (Metal)	6 (Dirt)	Average
SHDP	8.0(2.5)	13.8(6.1)	9.1(3.4)	3.5(1.3)	6.8(5.1)	5.0(3.8)	7.7(1.5)
SGRSNMF	8.8(1.0)	21.4(5.0)	7.3(1.3)	2.87(0.15)	2.5(4.6)	2.6(3.5)	7.6(1.0)
NMF-$\ell 1/2$	9.9(1.8)	25.9(6.0)	18.2(4.5)	3.3(1.1)	**2.5(0.7)**	3.9(1.9)	10.6(1.4)
SNSA	9.6(0.7)	15.9(3.9)	12.60(0.44)	3.9(0.1)	6.8(1.4)	5.2(1.0)	9.0(0.9)
uDAS	11.4(2.6)	19.2(6.3)	8.9(1.9)	3.6(0.5)	2.6(2.9)	10.0(8.8)	9.3(1.8)
DAEU	4.3(2.1)	5.1(4.7)	**4.1(3.1)**	1.9(1.2)	5.9(3.9)	8.0(3.4)	4.9(1.4)
STAEU 1 px	2.3(0.7)	6.6(3.6)	5.5(1.5)	**0.57(0.11)**	7.5(2.2)	2.1(1.3)	4.1(0.9)
MTAEU 9 px	**1.6(0.5)**	**4.8(3.1)**	6.4(2.1)	0.58(0.08)	7.2(1.9)	**1.0(0.4)**	**3.6(0.8)**

Table 3.7: MSE ($\times 10^{-2}$) between extracted abundance maps and the reference abundance maps for all methods for the Samson dataset. The figures are the mean of 25 runs along with the standard deviation. Best results are in bold typeface.

Method	1 (Soil)	2 (Tree)	3 (Water)	Average
SHDP	8.1(2.9)	6.2(1.5)	15.59(0.91)	10.0(1.2)
SGRSNMF	3.16(0.04)	5.74(0.32)	12.27(0.35)	7.06(0.22)
SNSA	12.5(2.1)	13.5(0.9)	16.91(0.87)	14.30(0.67)
uDAS	6.51(0.81)	6.66(0.54)	16.74(0.69)	9.97(0.38)
NMF-$\ell 1/2$	14.0(6.3)	10.5(2.0)	6.6(4.3)	10.3(3.3)
STAEU 1 px	2.04(0.79)	0.84(0.51)	1.0(0.2)	1.28(0.44)
MTAEU 9 px	**0.76(0.14)**	**0.37(0.07)**	**0.3(0.1)**	**0.48(0.08)**

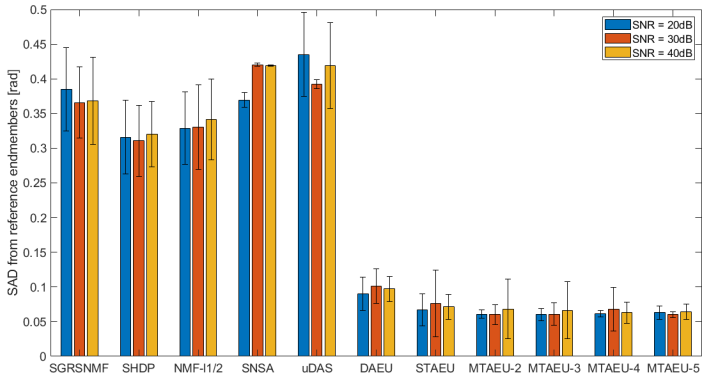


Figure 3.9: SAD from reference endmembers in radians for all methods and three different noise ratios for the Urban dataset (4 endmembers).

Tables 3.5, 3.7, and 3.6 show the MSE between the extracted abundance maps, both for individual endmembers and overall, for all blind unmixing methods for all the datasets. Again the numerical values are the mean value of 25 runs along with their standard deviation and the best results are in bold typeface. In all cases, the MTAEU method achieves the lowest average MSE score. It also has the least variance of all the methods. This is not surprising, both given the benefits of MTL discussed earlier, and the fact that the abundance maps for MTAEU are the mean of all the abundance maps of the k^2 autoencoders in the network. There will generally be some variance in the k^2 decoders which produce the abundances even though the decoders are all identical.

Fig. 3.8 shows an example of the abundance maps extracted by the method for the Urban dataset (4 endmembers) along with the reference maps. It can be seen that the abundance maps produced by the method are somewhat more intense and the fraction of low pixel abundances is lower in the maps, i.e., the method is more confident. This can be seen, e.g., in the figure for the Asphalt endmember. The roads are more intense in color than in the reference map shown. This is a result of using the nonlinear softmax function to enforce the ASC constraint. As the scaling factor α is increased, relatively small values get pushed to smaller values, increasing the "confidence" of abundances as they can be interpreted as probabilities being a result of the softmax function.

Fig. 3.9 shows the SAD from reference endmembers for all methods for three different SNR values, 20 dB, 30 dB, and 40 dB, for the Urban dataset with four endmembers estimated. As can be seen from the figure, all the methods are fairly robust to noise. Autoencoder methods are generally very robust to noise since they are denoising in nature as the network does not have the capacity to learn the added Gaussian noise. It is a bit surprising how well all the methods, except SGSRNMF and uDAS, perform with 20 dB SNR. Also, it is noticeable how small the variance for all noise levels is for the MTAEU method using 5×5 pixel neighborhood.

The experiments confirm all the main benefits of MTL that were listed in Sec. 3.1. Fig. 3.4 shows that the method is indeed spatial in nature and the multiple tasks speed up learning considerably. Fig. 3.3 along with figures 3.5, 3.7, and 3.6 show that the MTAEU method has less variance and better consistency than the DAEU and STAEU methods and that 3×3 neighborhoods are optimal, demonstrating that multitasking leads to better consistency and lower variance of endmember solutions. This is also true for the extracted abundance maps as our results also demonstrate. Lastly, the average running times for all methods for the Urban dataset can be seen in Table 3.8.

Table 3.8: Running times for all methods in minutes for the Urban dataset. The running time for the proposed method increases roughly linearly with increasing neighborhood size.

MTLAEU	STAEU	DAEU	SNSA	uDAS	SHDP	SGSNMF	NMF-ℓ1/2
1-6	1	2	37	16	2	240	2

Chapter 4

Convolutional Autoencoder For Spectral-Spatial Hyperspectral Unmixing

In this chapter, we present a new spectral-spatial linear mixture model and an associated estimation method based on a convolutional neural network autoencoder unmixing (CNNAEU). The CNNAEU technique exploits the spatial and the spectral structure of HSIs both for endmember and abundance map estimation. As it works directly with patches of HSIs and does not use any pooling or upsampling layers, the spatial structure is preserved throughout, and abundance maps are obtained as feature maps of a hidden convolutional layer. We compared the CNNAEU method to four conventional and three deep learning state-of-the-art unmixing methods using four real HSIs. Experimental results show that the proposed CNNAEU technique performs particularly well and consistently when it comes to endmembers' extraction and outperforms all the comparison methods.

4.1 Introduction

The high spectral resolution of hyperspectral sensors allows for the identification of distinct materials based on their spectral signatures. This is, however, complicated by the limited spatial resolution of such sensors. Because of the low spatial resolution, there is usually more than one distinct material in every pixel of the acquired image. Blind hyperspectral unmixing (HU) is the process of determining the spectral signatures of different materials and also their fractional abundances in every pixel. At the end of the process, we want to be able to reconstruct the original data with as little error as possible, from the set of extracted endmembers and their associated abundance maps.

4.1.1 Motivation and contributions

The motivation behind our method is to exploit the spatial structure of HSIs in addition to their spectral information. The pixels in HSIs are, as in other natural images, strongly correlated with their neighbors. This spatial structure contains information that can be used along with the spectral information to constrain further the unmixing problem.

Unmixing methods that exploit the spatial structure are called spectral-spatial methods. The most common way to utilize the spatial structure of HSIs is by refining a

spectral method by using some spatial prior to regularize or control the sparsity and the smoothness of the abundance maps obtained [52] or by doing preprocessing on the data prior to using a spectral only method. Developing entirely new spatially oriented methods is not as common.

In this chapter, a new spectral-spatial linear mixture model is introduced along with an associated estimation method based on a convolutional neural network (CNN) autoencoder. This is, to our best knowledge, the first blind spectral-spatial unmixing method based on CNNs. Since the method uses CNN it involves spatial filtering to extract features to aid it in the unmixing task, thus directly exploiting the spatial structure of the HSI. Furthermore, the technique preserves the spatial structure of HSIs throughout the method while abundance maps arise naturally as feature maps of a hidden convolutional layer. This makes it very simple to apply various spatial regularizations on the abundance maps, such as total variation (TV).

4.1.2 Publication review

The number of unmixing methods based on deep learning has been increasing rapidly over the last few years. Most of these methods are based on autoencoders and are therefore unsupervised. The nonnegative sparse autoencoder (NNSAE) has proved to be successful for unmixing. Such autoencoders satisfy the linear mixing model by having a linear decoder with nonnegative weights and by enforcing the abundance sum-to-one constraint (ASC) for abundances [123]. Examples of recent autoencoder methods are [12, 62, 66, 69, 77, 83]. All recent autoencoder approaches, except for [67], process a single spectrum at a time and hence do not include any spatial information.

Incorporating spatial information in unmixing, specifically for endmember extraction, has mainly been done in three different ways. Firstly, by developing new spatially oriented methods that provide some way of assessing the purity of a pixel or a neighborhood without using a geometrical method [124]. Examples of such methods are [125, 126]. Second, by the refinement of spectral only methods, e.g., by using spatial priors to regularize or penalize solutions obtained by existing methods. Good example of this is the SUnSAL-TV method [109] and also [110]. An example of methods using spectral–spatial weighted sparse regression is described in [111]. Examples of how a TV regularization term can be used to exploit spatial homogeneity in HSIs while keeping sharp edges are given in [44, 109, 112, 113]. Some examples of spatial regularization in NMF methods are [44, 114, 115]. An example of a hybrid linear and nonlinear NMF unmixing method based on a spatial prior is given in [116].

Lastly, incorporating spatial information can be done by preprocessing the data before applying a spectral only method. A good example of this is [127], where for each pixel, a scalar spatially derived factor relating to the similarity of pixels lying within a certain spatial neighborhood is computed and used to weigh the importance of the spectral information in a pixel in terms of its spatial context. Superpixel segmentation methods are also examples of methods that use a superpixel segmentation method as

Table 4.1: The indexing convention used for the matrices $\boldsymbol{M}_m$ and $\boldsymbol{W}_m$. Shown is an example for a 3×3 filter. We will often use the letter c for the center location, i.e., $\boldsymbol{W}_c = \boldsymbol{W}_5$ in this example.

	col. 1	col. 2	col. 3
row 1	$\boldsymbol{W}_1$	$\boldsymbol{W}_4$	$\boldsymbol{W}_7$
row 2	$\boldsymbol{W}_2$	$\boldsymbol{W}_5$	$\boldsymbol{W}_8$
row 3	$\boldsymbol{W}_3$	$\boldsymbol{W}_6$	$\boldsymbol{W}_9$

a preprocessing technique before unmixing. Examples of superpixel methods are given in [117–119] and [120] which applies a spatial group sparsity regularization derived from segmentation.

4.1.3 Notation

In this chapter, we write CONV n to refer to convolutional layer number n. We will use the notation $\boldsymbol{W}_{ij}, i, j = 1, \cdots, f$ to denote the $B \times R$ matrices that are obtained by fixing the first two indices of $\boldsymbol{F}$ to i and j. Often we will use a single index to index these matrices and will use the convention shown in Table 4.1 for labelling them.

The remainder of this chapter is organized as follows. Section 4.2.1 describes the proposed CNNAEU method. In Section 4.3, experimental results using four real HSI datasets are given along with a comparison with four conventional and three deep learning state-of-the-art HU methods. Finally, in Section 6.3 conclusions will be drawn.

4.2 Problem formulation and method

We are given P observed spectra, each having B bands. Having estimated the number of endmembers in the HSI as R, which can be done for example by using [28, 30], we assume a new spectral-spatial model

$$\boldsymbol{x}_p = \boldsymbol{M}\boldsymbol{s}_p + \sum_{i \in \mathcal{N}_p \setminus p} \boldsymbol{M}_i \tilde{\boldsymbol{s}}_i + \boldsymbol{\epsilon}_p, \quad p = 1, \cdots, P, \tag{4.1}$$

where $\boldsymbol{s}_p \in \mathbb{R}_+^{R \times 1}$ is an abundance vector that sums to one, $\boldsymbol{M} \in \mathbb{R}_+^{B \times R}$ contains the endmembers in its columns, $\mathcal{N}_p \setminus p$ is an $f \times f$ neighborhood around p, with f an odd number, while excluding p (the center pixel), $\boldsymbol{M}_i \in \mathbb{R}_+^{B \times R}$, $\tilde{\boldsymbol{s}}_i \in \mathbb{R}_+^{R \times 1}$, $\boldsymbol{M}_i \tilde{\boldsymbol{s}}_i$ is the spectral contribution to $\boldsymbol{x}_p$ from location i in its neighborhood, and $\boldsymbol{\epsilon}_p$ is noise. We index the pixels in the neighborhood $\mathcal{N}$ as shown in Table 4.1.

This new model differs from the widely used linear mixing model (LMM) by allowing all pixels in an $f \times f$ neighborhood centered on a pixel to contribute to the pixel's reconstruction. In this paper, we investigate the problem of estimating the endmember

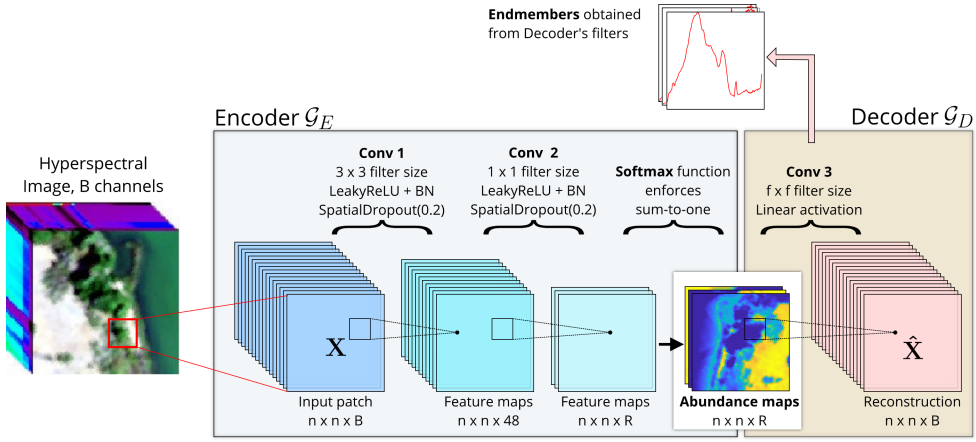


Figure 4.1: A schematic of the method. The convolutional autoencoder is trained on patches from the HSI. It has three convolutional layers. The abundance maps of the input patches are the feature maps from the second convolutional layer. The endmembers are extracted from the filters of the last layer which has linear activation. BN stands for batch normalization.

matrix $\boldsymbol{M}$ and the abundances $\boldsymbol{s}_p$ for every pixel in the HSI by interpreting the problem as blind unmixing, and solving it using a CNN autoencoder.

An autoencoder is a neural network that learns a representation of the HSI data. This means that the training is unsupervised, even though it might seem supervised, as it does not require labelled data. By introducing a hidden layer that acts as a bottleneck, the network learns a low-dimensional code for the input, and learn to reconstruct the input data from the code. The encoder is the part that performs this encoding, and the decoder is the part that reconstructs the input from the code. If the decoder is linear, the low-dimensional code for an HSI are the abundances and the weights of the decoder are the endmembers.

4.2.1 Estimation method

The method is a CNN autoencoder that is trained on N patches, $\mathcal{B}^i = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_{n\times n}\}$, $i = 1, \cdots, N$, and $\boldsymbol{x}_j \in \mathbb{R}^{B\times 1}$, from an HSI, $\boldsymbol{Y} \in \mathbb{R}^{D_1 \times D_2 \times B}$. We do not use biases, pooling layers or upsampling, only non-strided CONV layers that do not reduce the spatial resolution or the size of the input.

Fig. 4.1 shows a schematic of the method. The first layer in the network is the input layer. The second layer, CONV 1, is a 2D convolutional layer with 48 feature maps, filter of size 3×3, and LeakyReLU activation. We apply batch normalization [72] after the convolutional layer to speed up the learning. Next, we apply spatial dropout, a

version of dropout better suited to CNNs [80] to reduce over-fitting and improve generalization, with dropout rate 0.2. This means that entire feature maps are randomly set to zero with a probability of 20 percent.

Next comes another convolutional layer, CONV 2, having R feature maps, filter size 1×1, and LeakyReLU activation. Again we apply both batch normalization and spatial dropout with dropout rate 0.2 after the convolutional layer. This is the spectral bottleneck leading to the abundance maps. The next layer and the final layer in the encoding part of the autoencoder, is the layer that enforces the abundances sum-to-one constraint (ASC) at every pixel. It applies the softmax function pixel-wise to the collection of output feature maps from the previous layer, but scales the values by a factor α first. We used $\alpha = 3.5$ in experiments. The output of this layer are feature maps that can be interpreted as the abundances that sum-to-one at every pixel.

The last layer in the network is the linear decoder layer. This is a convolutional layer with B feature maps, filter size $f \times f$ where f is an odd number, and linear activation. This layer reconstructs the input patch from the abundance patches resulting from the abundances-sum-to-one enforcing layer. If the filter size is 1×1, it is easy to show that the weights, $\boldsymbol{W}$, of the filter is a $B \times R$ matrix which contains the endmembers as its columns. If the filter size is $f \times f$, we will get f^2 $B \times R$ matrices since the filter is an element of $\mathbb{R}^{f \times f \times R \times B}$. Furthermore, it is straightforward to rewrite the formula for the reconstruction of pixel p in a patch as

$$\hat{\boldsymbol{x}}_p = \sum_{m=1}^{f^2} \boldsymbol{W}_m \boldsymbol{h}_m,$$

where the single index m is indexing the locations in the $f \times f$ patch as shown in Table 4.1. Thus, for every location in row i and column j of the $f \times f$ neighborhood, there corresponds an element of $\mathbb{R}^{B \times R}$, i.e., a matrix having the same dimensions as the endmember matrix. This expression can be rearranged to obtain

$$\hat{\boldsymbol{x}}_p = \left(\sum_{m=1}^{f^2} \boldsymbol{W}_m \right) \boldsymbol{h}_c + \sum_{\substack{m=1 \\ m \neq c}}^{f^2} \boldsymbol{W}_m (\boldsymbol{h}_m - \boldsymbol{h}_c).$$

By comparing to ((4.1)) we can identify

$$\begin{aligned} \boldsymbol{M} &= \sum_{m=1}^{f^2} \boldsymbol{W}_m \\ \boldsymbol{M}_i &= \boldsymbol{W}_i \\ \boldsymbol{s}_p &= \boldsymbol{h}_c \\ \tilde{\boldsymbol{s}}_i &= \boldsymbol{h}_i - \boldsymbol{h}_c. \end{aligned} \tag{4.2}$$

Thus, the abundance maps are obtained by encoding the HSI with the autoencoder.

The endmembers are obtained directly from the weights of the decoder layer.

Loss function

We use the spectral angle distance (SAD) for the fidelity term of the loss function for our autoencoder. The spectral angle distance between two vectors, $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$, is given by

$$J_{\mathrm{SAD}}(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \arccos\left(\frac{\langle \boldsymbol{x}, \hat{\boldsymbol{x}} \rangle}{\|\boldsymbol{x}\|_2 \|\hat{\boldsymbol{x}}\|_2}\right). \tag{4.3}$$

The loss measures the discrepancy between the input patch and its reconstruction by the CNN. The loss for patch $\mathcal{B}^i$ is calculated as

$$\mathcal{L}^{(i)} = \frac{1}{|\mathcal{B}^i|} \sum_{\boldsymbol{x}_p \in \mathcal{B}^i} J_{\mathrm{SAD}}(\boldsymbol{x}_p, \hat{\boldsymbol{x}}_p). \tag{4.4}$$

The loss for the training dataset is then

$$\mathcal{L} = \sum_{i=1}^{N} \mathcal{L}^{(i)}. \tag{4.5}$$

4.3 Experiments

All experiments were performed using the Urban, Samson, Houson, and the Apex datasets and the extracted endmembers and abundance maps evaluated using 1.6 and 1.7, respectively.

The quality of extracted endmembers weighs more heavily in our evaluation of the method than the quality of the abundance maps. This is because the abundance maps produced by our method tend to be very binary, i.e., abundances are either very low or very high, almost like classification maps. This happens because of batch normalization and the fact that we are using a ReLU like activation and the softmax function to enforce the ASC constraint.

Another potential issue regarding the abundance estimation is our use of SAD loss. This loss, although very good for endmember extraction, is not an optimal metric for data reconstruction since it is scale invariant. It is possible that SAD loss leads to higher variance in the abundance estimation than is explainable by considering only the variance in the quality of the extracted endmembers. Still, methods using SAD loss can achieve fairly good results for abundance estimation as the results of the MTAEU method demonstrate.

We can always get fairly good abundance maps from good endmembers by, e.g., using the method of fully constrained least squares (FCLS) [128], or by using an autoencoder that has a fixed decoder, i.e., endmembers, and does not use batch normalization.

We did in fact additionally use a fully connected autoencoder to produce improved abundance maps for fixed endmembers extracted by CNNAEU.

To evaluate our method we compare the average scores for our extracted endmembers and abundance maps to the results from seven state-of-the-art methods. We use the mSAD and MSE of 25 runs for all experiments and report the average values along with the standard deviation. We also report the average SAD from reference along with standard deviation for each individual endmember. The comparison methods are listed in Table 4.2.

Table 4.2: The benchmark methods used for comparison in experiments.

Method	Description
$\ell_{1/2}$-NMF	Sparsity constrained nonnegative matrix factorization [39]. A spectral only method.
SGSRNMF	Spatial group sparsity regularized nonnegative matrix factorization [120]. A spectral-spatial method.
SHDP	Sticky hierarchical Dirichlet process [121]. A spectral-spatial method.
uDAS	Untied denoising autoencoder with sparsity [62]. A spectral only method.
SNSA	Stacked nonnegative sparse autoencoder [129]. A spectral only method.
DAEU	Deep autoencoder unmixing [66]. A spectral only method.
MTLAEU	Multitask learning autoencoder unmixing [67]. A spectral-spatial method.

Methods in rows 2, 3, and 7 in the table, are spectral-spatial methods, and the methods in rows 4-7 are based on autoencoders. The DAEU, MTAEU and CANNAEU methods were all initialized randomly, while all other methods are initialized or partially initialized using VCA.

4.3.1 Hyperparameter settings

Table 4.3 lists the CNNAEU's hyperparameters and their settings. Values of most hyperparameters were chosen using grid search and a single dataset (Urban) with reference endmembers. The size of the filters in layer CONV 1 was chosen to be 3×3 as larger sizes make the abundance maps look a bit fussy. The size of filters in layer CONV 2 was chosen to be 1×1 because of similar considerations. We performed a sensitivity analysis of selected hyperparameters w.r.t. the average SAD. One hyperparameter at a time was increased and decreased by 5%, 10%, 15%, and 20%, respectively. The experiment was performed using the Urban dataset, and ten runs were used to calculate the average SAD and MSE for each setting. The results, given in Table 4.4, show that the maximum deviation in average SAD is only 14% and the maximum

deviation in average MSE is only 17%. This means that the chosen settings of the hyperparameters do not correspond to some unusual and deep local minima. Values of the parameters that are close to the original values yield results that are close to the score obtained for the original values. Also, an increase of 14% in the average SAD from the reference score would not affect the ranking of our method for any datasets. Since the number of epochs is fixed, we allowed the number of patches used in training to vary between datasets as their sizes are different and we wish to avoid overtraining on smaller datasets. For our experiments, we determined the number of patches for the Urban dataset using grid search and then for the other datasets we scaled this number by the ratio of the product of the number of columns, rows, and bands for the dataset and the corresponding product for the Urban dataset.

Table 4.3: Hyperparameter settings used in the experiments.

Hyperparameter	Value used in experiments
Input patch size	40×40
Number of filters in CONV 1	48
Size of filters in CONV 1	3×3
Size of filters in CONV 2	1×1
Size of filters in CONV 3	11×11
Softmax scaling parameter α	3.5
Optimizer	RMSprop
Learning rate	0.0003
Batch size	15
Training Epochs	320

Table 4.4: Sensitivity to hyperparameter values. The numbers are the change in average SAD score when a hyperparameter is increased or decreased by the value displayed in the top row of the table. The numbers in parenthesis are the corresponding change in the average MSE score of the abundance maps. The average SAD score is the average SAD of extracted endmembers from reference endmembers of 10 runs for each hyperparameter configuration. The average MSE is the average MSE of extracted abundance maps from reference maps of 10 runs for each hyperparameter configuration.

Hyperparam.	Batch size	Learning rate	#patches	#filters in CONV 1	Patch size
-5%	7.4% (5.5%)	3.3% (7.2%)	7.7% (3.4%)	6.4% (5.2%)	6.6% (1.6%)
+5%	2.6% (2.5%)	8.9% (10.8%)	5.4% (4.5%)	5.5% (1.3%)	4.5% (-0.5%)
-10%	10.7% (3.5%)	7.2% (3.9%)	6.2% (11.2%)	9.0% (2.4%)	11.2% (1.1%)
10%	6.2% (2.4%)	12.2% (7.0%)	14.2% (12.2%)	10.0% (-0.5%)	9.4% (5.9%)
-15%	13.6% (12.4%)	4.4% (2.0%)	7.7% (16.6%)	8.3% (4.7%)	9.8% (11.2%)
+15%	5.8% (5.5%)	11.7% (5.8%)	15.5% (12.2%)	10.1% (8.6%)	17.1% (8.8%)
-20%	3.5% (10.2%)	-1.4% (8.9%)	10.8% (10.7%)	13.5% (12.5%)	4.3% (11.9%)
+20%	7.5% (10.0%)	12.5% (11.7%)	10.0% (6.9%)	13.7% (5.7%)	4.9% (8.4%)

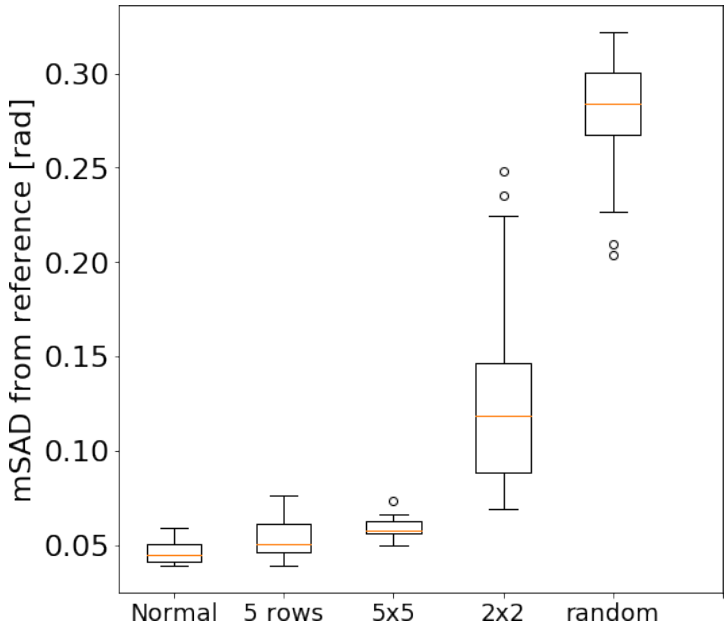


Figure 4.2: The effect of gradual spatial degradation of the input HSI on the performance of the method.

4.3.2 Spatiality

To confirm if the method is spatial in nature, we performed an experiment where we investigate the performance of the method with increasing spatial degradation of an HSI. If the technique is utilizing the spatial structure of the HSI to perform the unmixing, then gradually destroying the spatial structure by permuting areas in the HSI should result in the method performing increasingly worse when holding all hyperparameters constant.

Fig. 4.2 shows the result of this experiment for the Urban dataset as a boxplot. There are 25 runs behind every setting. We used patch size of 40×40 with 200 training patches in the training dataset and trained the network for 100 epochs. The categorical labels have the following meaning:

Normal This is using the original dataset unchanged.

5 rows This dataset is obtained by permutation of the rows of the image in blocks of five.

5 x 5 This dataset is obtained by dividing the original image into 5 by 5 patches and randomizing their locations.

2 x 2 This dataset is obtained by dividing the original image into 2 by 2 patches and randomizing their locations.

Random This dataset is obtained by totally randomizing every pixel location by a random permutation of the array of spectra before reshaping it into a 2D image.

As can be seen clearly from the figure, the performance, as measured by the mSAD in radians, decreases with increasingly fine grained permutations of the pixels of the original image. When all spatial structure has been completely destroyed, the mSAD is between 0.25 and 0.30 radians after 100 epochs compared to around 0.05 radians for the original image. This shows that the CANNAEU method is indeed spatial in nature.

4.3.3 The matrices $\boldsymbol{W}_{ij}$

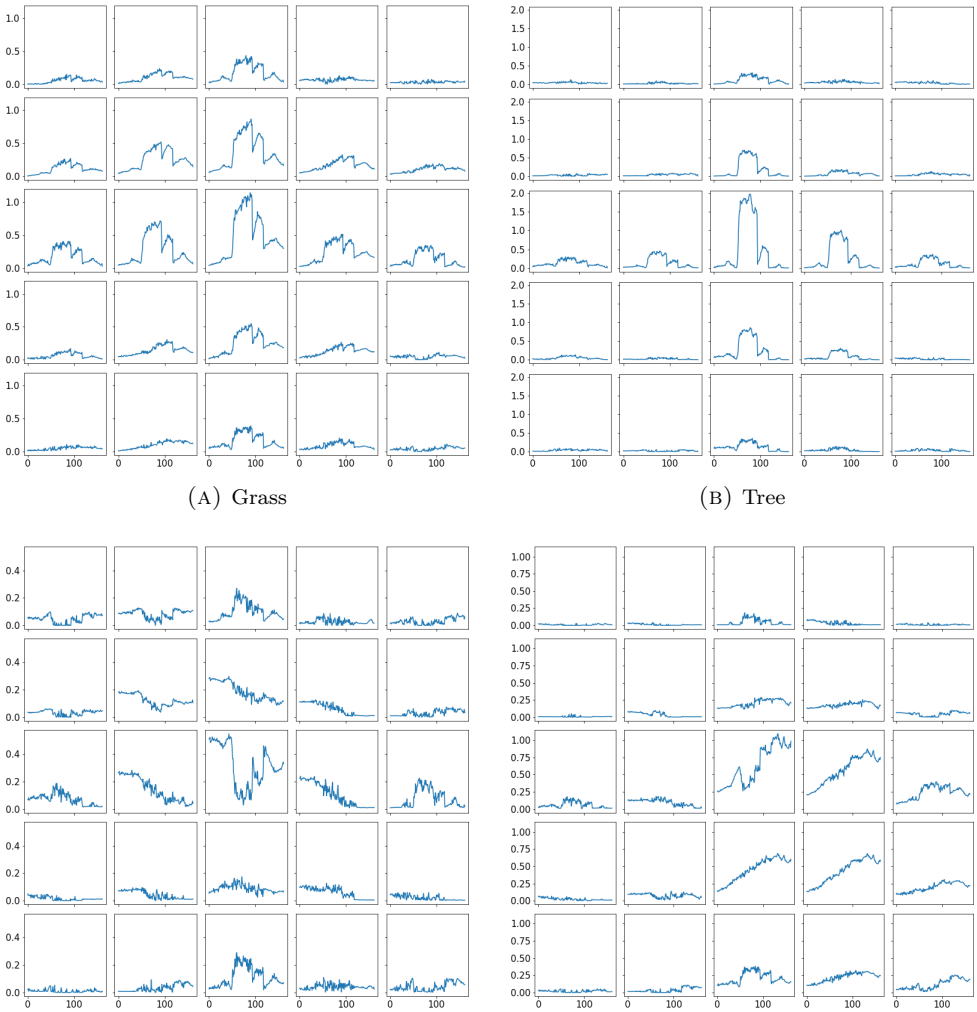
In this section, we will investigate the matrices $\boldsymbol{W}_{ij}$, $i, j = 1, \cdots, f$, that are obtained from the decoder's filter by fixing the first two indices to i, j. Fig. 4.3 shows plots of the contents of the filter for the Urban dataset for all four endmembers and all 25 locations for a filter having size 5×5.

Looking at Fig. 4.3, there are two things that stand out. Firstly, the relative scale of the plots compared to the center plot, and lastly the variations in the shape of the plots, the filter for certain endmembers seems to contain plots of more than one different endmembers. Based on our experimentation, we can summarize the behavior of the filter regarding the two above mentioned outstanding features with regard to the spatial structure of the HSI as following:

1. The relative scale of surrounding locations is entirely dependent upon the amount of spatial correlations in an HSI. If there are no spatial correlations, only the center location will be non zero and it will contain the estimated endmember.
2. If an endmember is often considerably mixed with another, this will be reflected in the filter plots of the endmember. The plots in locations other than the central location can look similar to the other endmember.

Fig. 4.4 illustrates observation number 1 here above. When the spatial structure of the Urban HSI has been destroyed by randomly permuting the pixels, the scale of the plots in non-central locations goes to zero as can be seen in Fig. 4.4b. However, by selecting a sub-image of Urban that is more homogeneous with respect to the Tree and Grass endmembers than the original image, the plots in Fig. 4.4c are obtained. The relative scale of non-central locations has increased with the increased spatial correlations in the scene.

Fig. 4.3c illustrates observation number 2 above well. It is the filter plot for the Roof endmember which, in the Urban HSI, is very often mixed with the Trees and Grass endmembers. In locations (1,3), (3,1), (5,3) and (3,5) the plots contain nothing that



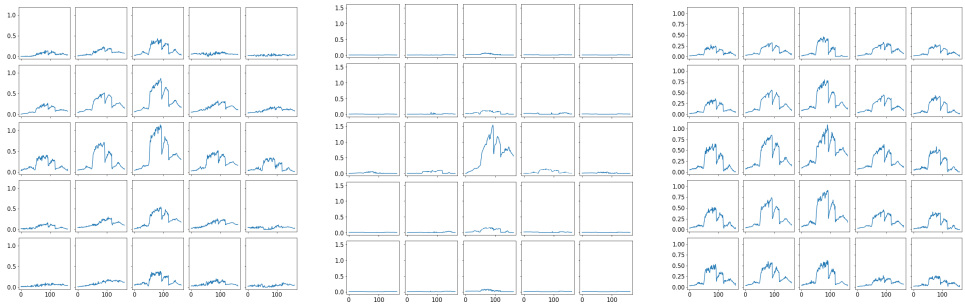
(A) Grass

(B) Tree

(C) Roof

(D) Asphalt

Figure 4.3: Plots of the endmember matrices of the 5x5 filter for the Urban dataset.



(a) Original

(b) Randomized. No spatial correlations.

(c) More homogenous subimage.

Figure 4.4: The "Grass" endmember filter for different degrees of spatial correlations in the Urban dataset.

resembles the Roof endmember, but looks very similar to the Tree/Grass endmembers. The same can also be observed in the plot for the Asphalt endmember in Fig. 4.3d, which is also often mixed with vegetation endmembers in the Urban HSI. The same locations show something that looks similar to the vegetation endmembers.

The spatial nature of the reconstruction of a pixel's spectrum by the CNN autoencoder as described by (4.1) makes the two observations above very reasonable. For the case with no spatial structure, then generally $\boldsymbol{h}_m$ is very different from $\boldsymbol{h}_c$, while at the same time the network cannot rely on the neighboring abundances of a pixel for its reconstruction, forcing it to make the $\boldsymbol{W}_m$ close to zero for locations other than the center.

Now, let us consider the case of an HSI that is heavily correlated. In this case the network can rely on neighboring pixels for the reconstruction of a pixel. However, we can generally expect $\boldsymbol{h}_m$ be close to $\boldsymbol{h}_c$ so the their difference will be small in magnitude, forcing the network to increase the norms of $\boldsymbol{W}_m$ for locations other than the center. This means that the relative size of the plots of $\boldsymbol{W}_{ij}$ will become similar and large compared to the center location, as Fig. 4.4c is showing.

It is clear that a decoder filter of size $f \times f$ can give some qualitative information about the HSI being unmixed. Also, having f larger than 1 does indeed give slightly better estimation of endmembers.

4.3.4 Endmembers

In this section, we will focus on the extracted endmembers from the experiments. Table 4.5 shows the mSAD of the extracted endmembers along with standard deviation from 25 runs, along with the average SAD from references for individual endmembers for the Urban dataset for all the methods. Fig. 4.5 shows the extracted endmembers

for all the methods for the Urban dataset with four reference endmembers. The blue curves are the extracted endmembers from 25 runs while the red curves are the reference endmembers.

As can be seen from the figure, the MTAEU and CNNAEU methods are in a class of their own for this dataset. They are the only methods that have both good performance and excellent consistency. The proposed method actually has better SAD score and less variance of the two.

The SNSA method has a very low variance also, but its solution is not good. All the methods except for the DAEU, MTAEU and CNNAEU have trouble with estimating the Grass endmember correctly, which is a bit surprising since it is a very well represented endmember in the dataset. The NMF-$\ell_{1/2}$ does the best job aside from the aforementioned methods.

Table 4.6 shows the mSAD of the endmembers extracted by all the methods for the Samson dataset along with the average SAD from references for individual endmembers. Fig. 4.6 shows the extracted endmembers along with the reference. It can be seen from the figure that most of the methods do not have much trouble with this dataset. The method that has the best SAD score is the MTAEU method and it also has the second least variance. This is the only dataset where we had to use small patch size in order to get good and stable results. The dataset itself is also very small, only 95 by 95 pixels, which could be the reason why a higher number of small patches gives better results than having fewer and much larger patches in the training dataset.

Table 4.5: Urban dataset. Mean SAD and standard deviation comparison for endmember extraction in radians. Best results in bold.

Method	Asphalt	Grass	Tree	Roof	Average
NMF-$\ell_{1/2}$	0.1161±0.0233	0.573±0.323	0.1571±0.0921	0.4891±0.2065	0.3338±0.0644
SGSRNMF	0.2446±0.0204	1.3006±0.0444	0.0967±0.0113	0.1916±0.0862	0.4584±0.0148
SHDP	0.2658±0.0751	0.5524±0.3172	0.0777±0.0171	0.4117±0.172	0.3269±0.0555
SNSA	0.2912±**0.0003**	1.2165±**0.0042**	0.0777±**0.0001**	0.0923±**0.0005**	0.4194±**0.0011**
uDAS	0.1805±0.0321	1.1609±0.1515	0.1496±0.0385	0.17±0.0011	0.4153±0.0403
DAEU	0.072±0.0235	0.0769±0.0285	0.0707±0.0291	0.2155±0.0845	0.1088±0.0234
MTAEU	0.0843±0.0046	0.0421±0.0036	0.0539±0.0039	0.0415±0.0045	0.0555±0.0019
CNNAEU	**0.0575**±0.0058	**0.0366**±0.0047	**0.0321** ±0.0039	**0.0332**±0.0066	**0.0398** ±0.003

Table 4.6: Samson dataset. Mean SAD and standard deviation comparison for endmember extraction in radians. Best results in bold.

Method	Soil	Tree	Water	Average
NMF-$\ell_{1/2}$	0.4013±0.2097	0.0636±0.0379	0.1319±0.1005	0.1989±0.1034
SGSRNMF	0.0086±**0.0001**	0.0395±0.0019	0.0923±**0.0024**	0.0468±**0.0003**
SHDP	0.2147±0.3299	0.0375±**0.0004**	0.2064±0.0916	0.1527±0.139
SNSA	0.2493±0.3122	0.075±0.0146	0.2844±0.0677	0.2029±0.1218
uDAS	0.0312±0.0014	0.0547±0.0049	0.1405±0.0117	0.0755±0.0051
DAEU	0.0497±0.0216	0.0553±0.0121	0.053±0.0208	0.0527±0.0117
MTAEU	**0.0225**±0.006	**0.0371**±0.0028	**0.0338**±0.0031	**0.0311** ±0.0017
CNNAEU	0.0373±0.021	0.0397±0.0038	0.043±0.0092	0.04±0.0067

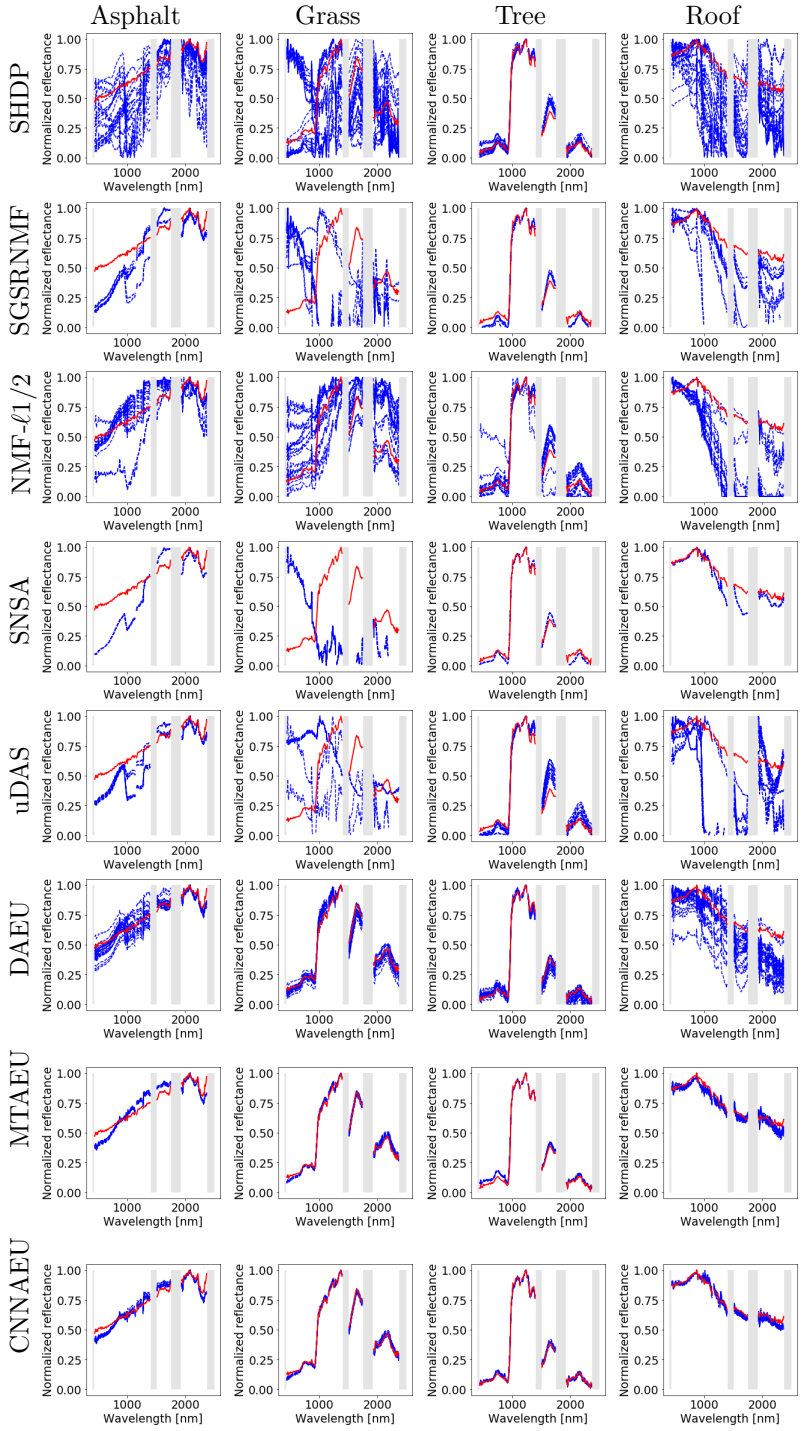


Figure 4.5: Urban dataset. Endmembers extraction for comparison. Reference endmembers in red.

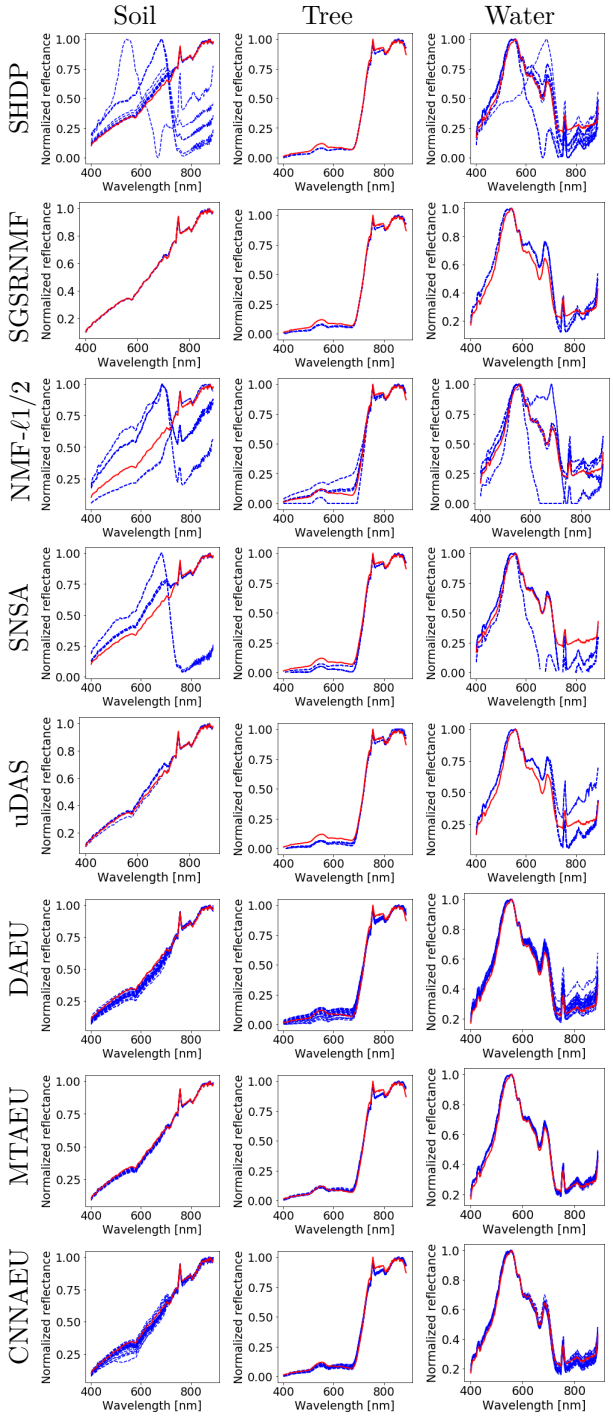


Figure 4.6: Samson dataset. Endmembers extraction for comparison. Reference endmembers in red.

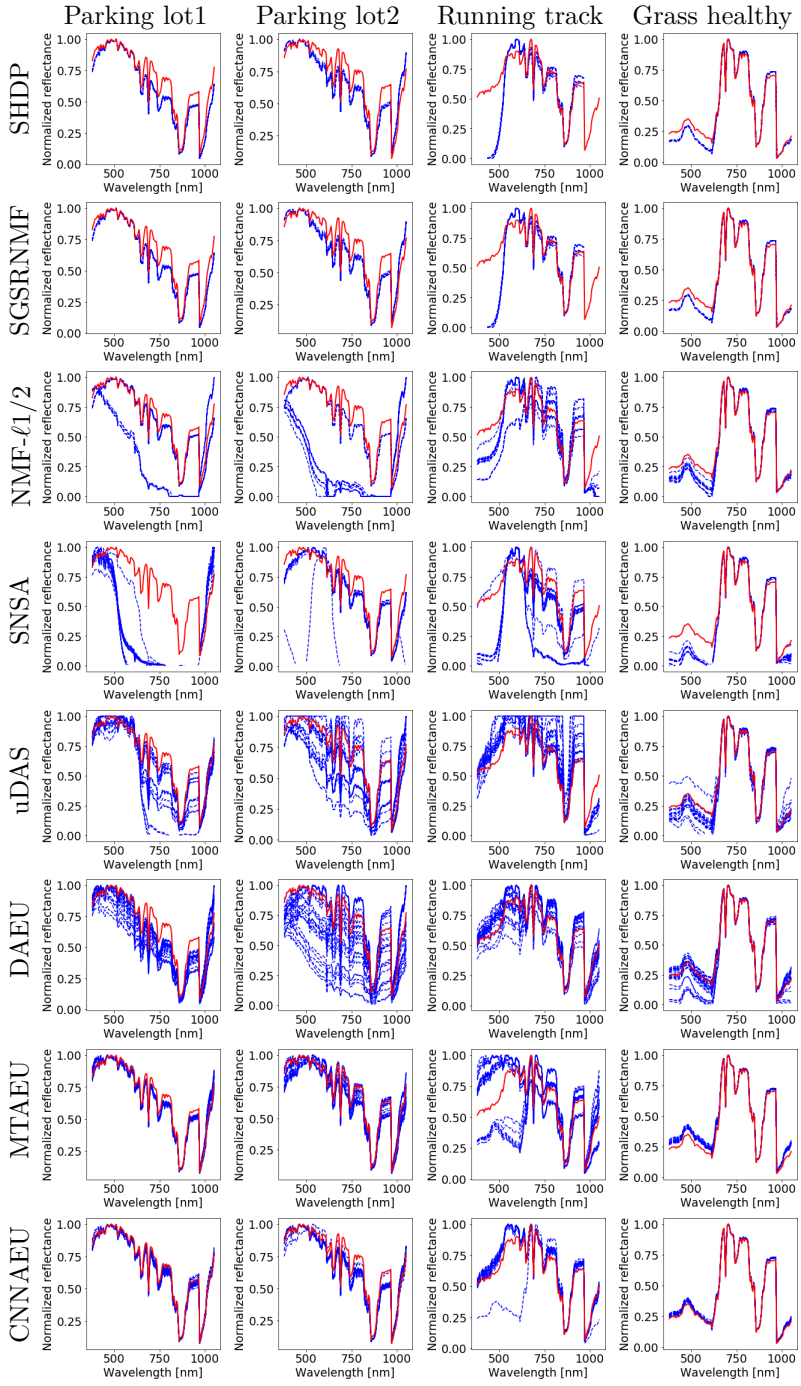


Figure 4.7: Houston dataset. Endmembers extraction for comparison. Reference endmembers in red.

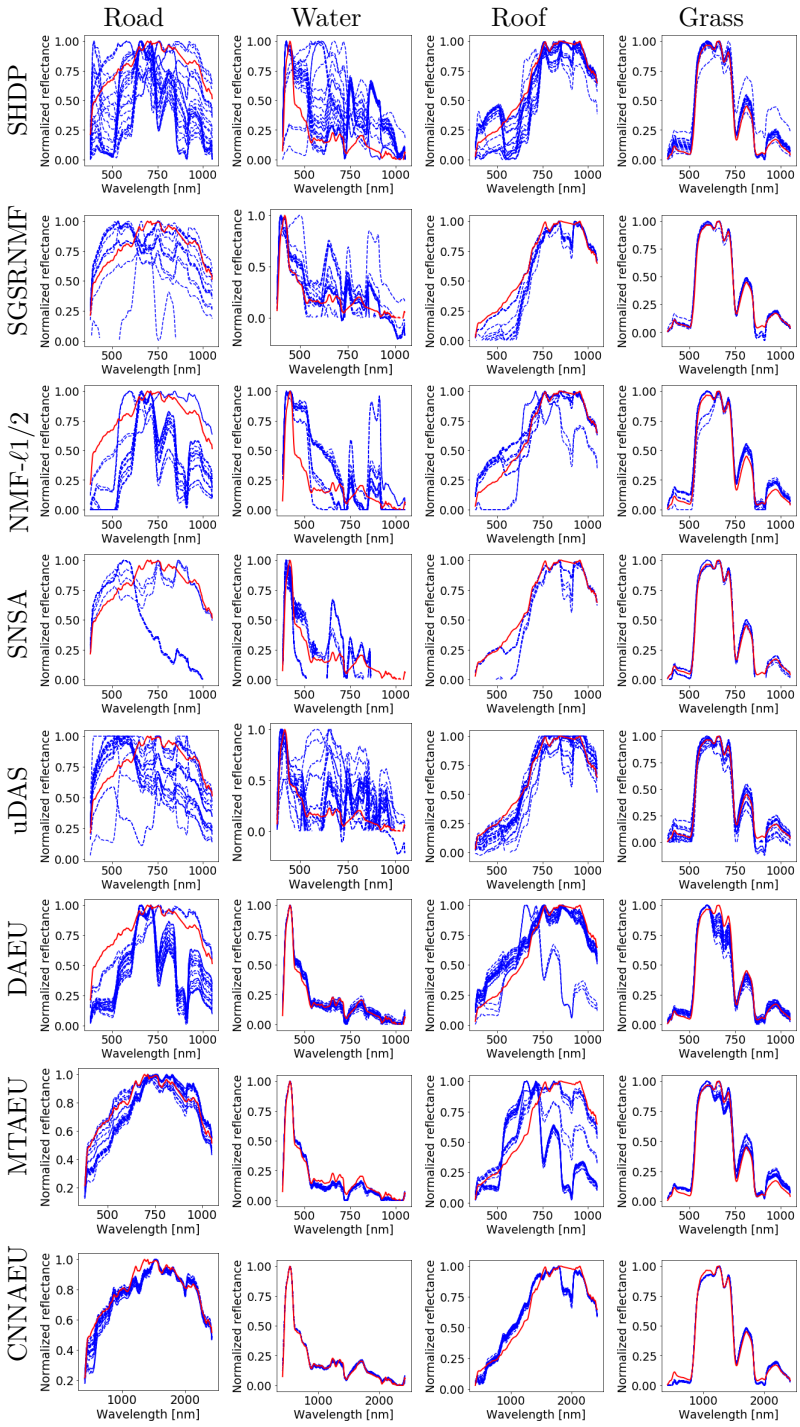


Figure 4.8: Apex dataset. Endmembers extraction for comparison. Reference endmembers in red.

Table 4.7: Houston dataset. Mean SAD and standard deviation comparison for endmember extraction in radians. Best results in bold.

Method	parking_lot1	parking_lot2	running_track	grass_healthy	Average
NMF-$\ell_{1/2}$	0.0665±0.0029	0.7426±0.091	0.2678±0.0696	0.1366±0.0252	0.3034±0.0305
SGSRNMF	0.0939±**0.0040**	0.1347±**0.0018**	0.4289±**0.0034**	0.0931±**0.0010**	0.1877±**0.0010**
SHDP	0.0751±0.0024	0.6988±0.0255	0.2323±0.0013	0.1585±0.0283	0.2912±0.0082
SNSA	0.0971±0.0953	0.8269±0.0366	0.4958±0.1987	0.2485±0.0174	0.4171±0.0568
uDAS	0.113±0.11	0.3039±0.1492	0.155±0.053	0.108±0.056	0.17±0.0388
DAEU	0.1263±0.0352	0.2798±0.1588	0.114±0.0389	0.0937±0.0825	0.1534±0.0353
MTAEU	0.0427±0.009	0.0786±0.0128	0.2894±0.0886	0.0642±0.0104	0.1187±0.0226
CNNAEU	**0.0291** ±0.0095	**0.0775**±0.007	**0.0671**±0.0767	**0.0272** ±0.0062	**0.0502**±0.0193

Table 4.8: Apex dataset. Mean SAD and standard deviation comparison for endmember extraction in radians. Best results in bold.

Method	Road	Water	Roof	Grass	Average
NMF-$\ell_{1/2}$	0.5265±0.0484	0.4725±0.0717	0.1301±0.0175	0.1036±0.0147	0.3081±0.027
SGSRNMF	0.1779±0.1707	0.3619±0.0917	0.1327±0.0794	0.0574±0.0082	0.1825±0.0788
SHDP	0.4123±0.1293	0.6068±0.1956	0.196±0.0631	0.0947±0.027	0.3274±0.069
SNSA	0.6324±0.179	0.5669±0.0909	0.0946±0.0604	0.0607±0.0024	0.3387±0.0525
uDAS	0.3109±0.1021	0.5195±0.1595	0.1289±0.0483	0.0984±0.0469	0.2644±0.0623
DAEU	0.3637±0.1113	0.1061±0.0245	0.232±0.1731	0.1193±0.0369	0.2053±0.0343
MTAEU	0.1126±0.0403	0.1365±0.0163	0.546±0.2461	0.0992±0.0076	0.2236±0.0711
CNNAEU	**0.0587±0.0196**	**0.0417±0.0037**	**0.1233±0.0060**	**0.0621±0.0038**	**0.0714±0.0045**

Table 4.7 shows the mSAD of the endmembers extracted by all the methods for the Houston dataset along with the average SAD from references for individual endmembers. Fig. 4.7 shows all the extracted endmembers along with the reference. Looking at the figure, one can see that the proposed method has again both very good endmembers and very good consistency and low variance. Only in one run does it fail to extract the "running_track" endmember correctly and instead produces a variant of the "grass_healthy" endmember. The MTAEU method has some trouble with the consistency of the endmember "running_track" and often finds a variant of the "grass_healthy" endmember instead, most likely it is the "grass_stressed" endmember. The DAEU method has high variance for this dataset and often gets a bad solution for the "parking_lot2" endmember. It is very correlated to the "parking_lot2" endmember which could be an explanation for this behavior. The NMF methods generally have trouble with the "running_track" endmember.

Table 4.8 shows the mSAD of the extracted endmembers for all the methods for the Apex dataset along with the average SAD from references for individual endmembers. Fig. 4.8 shows all the extracted endmembers along with the reference. Here the proposed method achieves the best SAD score and has the best consistency and lowest variance by far of all the methods. The MTAEU method often incorrectly estimates the roof endmember and extracts an endmember corresponding to trees. There are a lot of trees in the image and often obscuring roofs partially so there are a lot of mixed pixels containing both the tree and the roof endmember.

The DAEU method has less trouble with the roof endmember but is often estimating the tree endmember instead of the road endmember. There are also a lot of mixed pix-

els containing both these endmember in the scene. It is somewhat surprising how much trouble all the other methods, uDAS in particular, have with the water endmember. This endmember is abundant in the scene and its variability is not great.

Regarding the quality of extracted endmembers and method consistency it is clear that the proposed method ranks first among all the methods in the experiments. It seems that the spatial aspect of the method greatly reduces the variance of the method and improves its consistency. It outperforms the MTAEU method, which is also a spectral-spatial exhibiting low variance and good consistency, in all the datasets except for Samson.

4.3.5 Abundance maps

As was mentioned at the start of this section, the abundance maps produced by CNNAEU are very intense looking, i.e., both very sparse and binary. It is possible to extend the method so it refines the abundance maps after extracting the endmembers. This could be done by using an autoencoder in serial with the unmixing one, which has its decoder's weights set to be the extracted endmember matrix and made non-trainable so they are fixed, and does not use batch normalization layers. By training it for a few epochs, it will produce abundance maps which are not nearly as binary looking.

We did this to improve the abundance map results, using a fully connected autoencoder with a single hidden layer and enforcing the ASC constraint using the softmax function. The resulting method is named CNNAEU2 and we will report MSE from the reference abundance maps for both CNNAEU and CNNAEU2.

In Table 4.9 the mean MSE and standard deviation comparison for abundance maps extraction for the Urban dataset with four endmembers is given. Fig. 4.9 shows examples of the extracted abundance maps by all the methods for the same dataset. The figure shows well the issue with the abundance maps produced by the proposed method. It also shows the refined maps by the CNNAEU2 method.

Table 4.9: Urban dataset. Mean MSE and standard deviation comparison for abundance maps extraction. Best results in bold.

	Asphalt	Grass	Tree	Roof	Average
NMF-$\ell_{1/2}$	0.1733±0.0013	0.2035±0.0039	0.1886±0.0032	0.0572±0.0006	0.1557±0.0004
SGSRNMF	0.0816±0.0058	0.1995±0.0103	0.0715±0.0171	0.0358±0.0018	0.0971±0.0085
SHDP	0.0909±0.0144	0.148±0.0724	0.0833±0.0235	0.0745±0.0569	0.0992±0.0204
SNSA	0.1129±**0.0005**	0.1681±**0.0005**	0.126± **0.0007**	0.0433±**0.0002**	0.1126±**0.0002**
uDAS	0.1013±0.0036	0.1872±0.0381	0.0916±0.0103	0.0385±0.0012	0.1046±0.0071
DAEU	0.0307±0.0172	0.0556±0.0258	0.0391±0.033	0.0189±0.0131	0.036±0.0147
MTAEU	0.023±0.001	0.0225±0.0016	**0.0068** ±0.0009	0.0079±0.0008	0.015±0.0008
CNNAEU	0.066±0.0041	0.0868±0.0071	0.0437±0.0073	0.0284±0.0067	0.0562±0.0049
CNNAEU2	**0.0156** ±0.0016	**0.0158** ±0.0016	0.0073±0.0015	**0.0073** ±0.0015	**0.0115** ±0.001

Table 4.10: Samson dataset. Mean MSE and standard deviation comparison for abundance maps extraction. Best results in bold.

	Soil	Tree	Water	Average
NMF-$\ell_{1/2}$	0.14±0.0622	0.1045±0.0192	0.0658±0.0417	0.1034±0.0323
SGSRNMF	0.0316±**0.0004**	0.0574±0.0031	0.1227±0.0034	0.0706±0.0021
SHDP	0.0814±0.0288	0.0623±0.0145	0.1559±0.009	0.0999±0.012
SNSA	0.1251±0.02	0.1347±0.0084	0.1691±0.0086	0.143±0.0066
uDAS	0.0651±0.0079	0.0666±0.0053	0.1674±0.0067	0.0997±0.0037
DAEU	0.0439±0.045	0.0198±0.0103	0.0239±0.0471	0.0292±0.0311
MTAEU	**0.0076**±0.0014	**0.0037±0.0007**	**0.0029± 0.0010**	**0.0048±0.0008**
CNNAEU	0.1068±0.06	0.0899±0.0558	0.0376±0.0982	0.0781±0.0576
CNNAEU2	0.0765±0.0676	0.0631±0.0707	0.0166±0.0133	0.0521±0.0449

Table 4.11: Houston dataset. Mean MSE and standard deviation comparison for abundance maps extraction. Best results in bold.

Method	parking_lot1	parking_lot2	running_track	grass_healthy	Average
NMF-$\ell_{1/2}$	0.1366±0.0163	**0.0997** ±0.0055	0.0183±0.0039	**0.0059** ±0.0027	0.0651±0.005
SGSRNMF	0.2799± **0.0003**	0.2253±**0.0004**	0.0359±**0.0009**	0.004± **0.0001**	0.1363± **0.0002**
SHDP	0.1881±0.0079	0.1362±0.0049	0.0214±0.0012	0.0085±0.0007	0.0885±0.0027
SNSA	0.2455±0.0038	0.1554±0.0125	0.0629±0.0172	0.1235±0.0117	0.1468±0.0025
uDAS	0.2428±0.0459	0.2218±0.0394	0.0319±0.0051	0.0452±0.0794	0.1354±0.0245
DAEU	0.1903±0.0461	0.1665±0.0692	0.0834±0.0795	0.0266±0.0195	0.1167±0.0335
MTAEU	0.1558±0.0821	0.2431±0.0968	0.161±0.1094	0.0151±0.0101	**0.1437** ±0.0537
CNNAEU	0.2178±0.0685	0.2034±0.0701	0.0669±0.0153	0.0314±0.0111	0.1299±0.0348
CNNAEU2	0.1196± **0.0926**	0.1187±0.081	**0.0090**±0.0064	0.0067±0.0011	**0.0635**±0.0441

Table 4.12: Apex dataset. Mean MSE and standard deviation comparison for abundance maps extraction. Best results in bold.

Method	Road	Water	Roof	Grass	Average
NMF-$\ell_{1/2}$	0.2926±0.0385	0.1578±0.0121	0.051±0.0152	0.0199±0.0226	0.1303±0.0096
SGSRNMF	0.1027±0.0674	0.1646±0.0231	0.0461±0.0086	**0.0084** ±0.0224	0.0805±0.0255
SHDP	0.172±0.046	0.127±0.0302	**0.0363** ±0.0106	0.0204±0.0214	0.0889±0.018
SNSA	0.0973± **0.0016**	0.1862±0.0072	0.0809±**0.0003**	0.1091± **0.0052**	0.1184±**0.0026**
uDAS	0.1082±0.0801	0.2319±0.0401	0.0567±0.0056	0.0284±0.0233	0.1063±0.0254
DAEU	0.183±0.0226	0.1074±0.0054	0.0809±0.0486	0.0443±**0.0052**	0.1039±0.0094
MTAEU	0.1457±0.0523	**0.0790±0.0048**	0.1267±0.0543	0.0432±0.014	0.0987±0.0242
CNNAEU	0.1458±0.0196	0.1058±0.0181	0.0959±0.0185	0.0647±0.007	0.1031±0.0111
CNNAEU2	**0.0575** ±0.0339	0.0808±0.0098	0.0448±0.0142	0.0278±0.0055	**0.0527**±0.0142

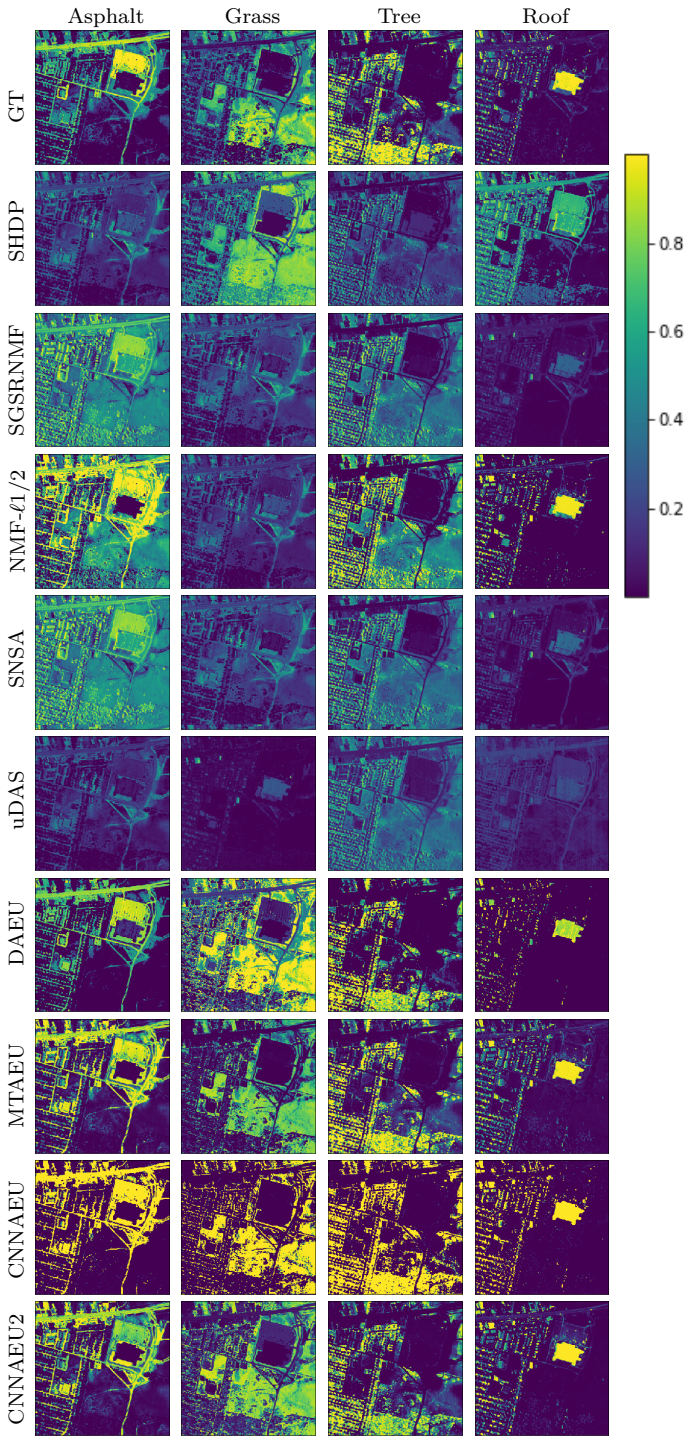


Figure 4.9: Urban dataset. Abundance maps extraction comparison and reference maps in the first column.

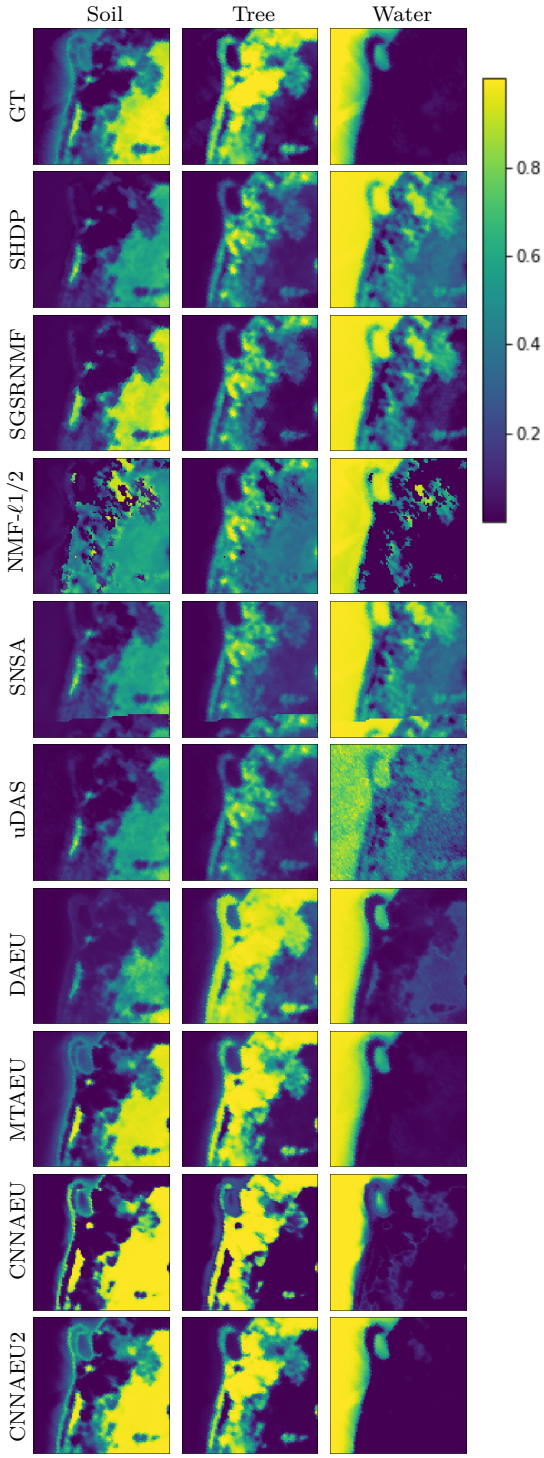


Figure 4.10: Samson dataset. Abundance maps extraction comparison and reference maps in the first column.

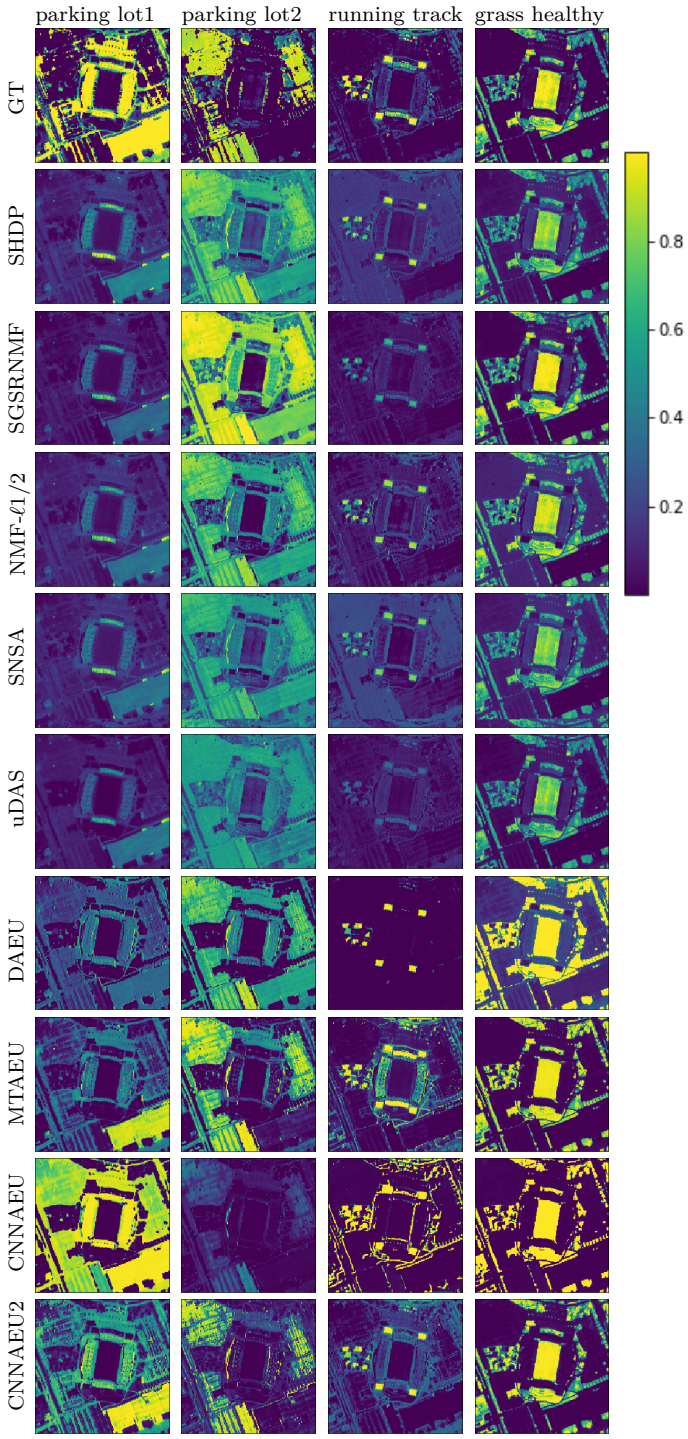


Figure 4.11: Houston dataset. Abundance maps extraction comparison and reference maps in the first column.

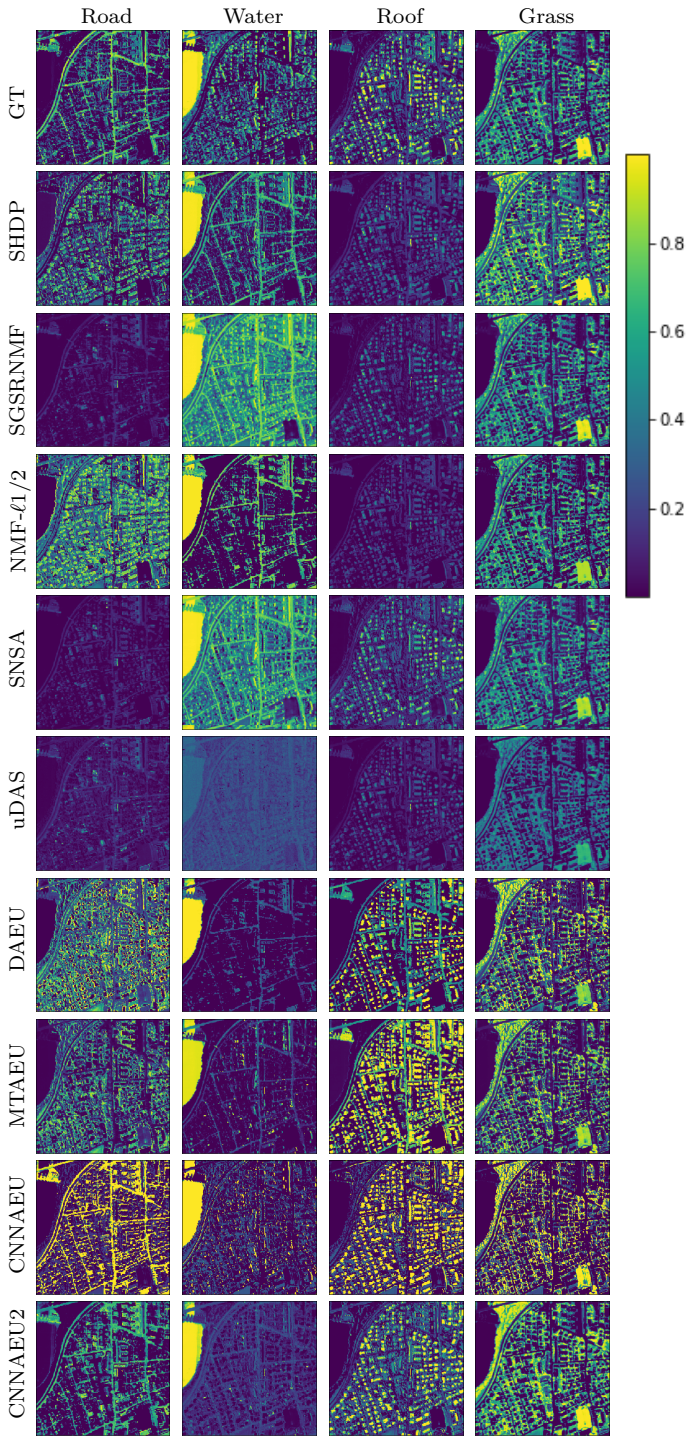


Figure 4.12: Apex dataset. Abundance maps extraction comparison and reference maps in the first column.

The MTAEU method achieves the lowest MSE score with a very low variance, and is closely followed by the CNNAEU2 method which also has very small variance. The maps produced by CNNAEU rank third (excluding CNNAEU2). As can be seen from the figure, even though they are intense, they are still good maps and could actually be used for classification.

Table 4.10 shows the MSE from reference abundance maps for the Samson dataset for all the methods. Fig. 4.10 shows examples of the extracted abundance maps by all the methods for the same dataset. MTAEU scores best of all the methods again followed closely by CNNAEU2. It is interesting to see how high the variance is for the CNNAEU method. It does not reflect the low variance of the SAD score for the extracted endmembers. Of the other spectral-spatial methods, SGSRNMF achieves the best score.

Table 4.11 shows the MSE of abundances from the reference abundance maps for the Houston dataset for all the methods. Fig. 4.11 shows examples of the extracted abundance maps by all the methods for the Houston dataset. On this dataset it is CNNAEU2 that achieves the best score. Second best is NMF-$\ell_{1/2}$ and third best is DAEU. The SHDP and SGSRNMF and the SNSA methods have all very low variance compared to the other methods.

Table 4.12 shows the MSE from the reference abundance maps for the Apex dataset for all the methods. Fig. 4.12 shows examples of the extracted abundance maps by all the methods for the Apex dataset. The maps produced by the CNNAEU2 method achieve the lowest MSE score for this dataset. All the other methods have average MSE close to 0.1.

4.3.6 Robustness to noise

To investigate the method's robustness to noise, we performed experiments where we added white noise with SNR = 20 dB, 30 dB and 40 dB to the Urban dataset prior to unmixing with all methods. Fig. 4.13 shows a bar chart of the mSAD score for all the methods versus SNR grouped by methods. The figure shows that all the methods are fairly robust to noise. Also. it is surprising how well the methods, except for SGSRNMF and uDAS, performed with 20 dB SNR. The proposed method seems to be largely unaffected by the added noise. The low variance of the method for all noise levels stands out in the comparison.

4.3.7 Computational complexity

Here, we discuss the running time of the method and compare it to the running time of the other methods. Table 4.13 shows the average running time in seconds for all the methods for all the images and additionally one large image, a 900x900 pixel subset of the Apex HSI named "Apex big" in the table. It must be kept in mind that for the

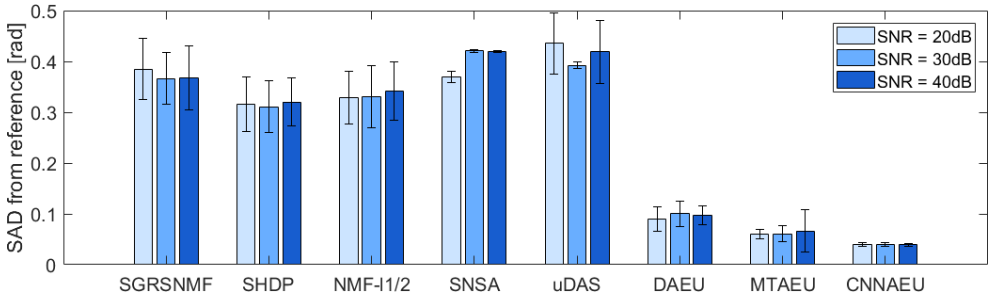


Figure 4.13: Urban dataset. Mean SAD and standard deviation comparison in radians for robustness test.

Table 4.13: CPU time in seconds for all unmixing methods and all datasets used in the experiments along with one additional large dataset named "Apex big" in the table.

Method	Urban	Samson	Houston	Apex	Apex big
$\ell_{1/2}$-NMF	83	7	31	152	1186
SGSNMF	276	27	85	312	1007
SHDP	14425	939	8745	16743	NA
SNSA	809	106	708	1813	5856
uDAS	940	5	158	1654	14424
DAEU	226	4	65	165	242
MTAEU	80	58	105	95	223
CNNAEU	143	37	59	111	150

methods DAEU, MTAEU, and CNNAEU, the training set is only a small subset of all possible patches or pixels, so the effect of the image size is not very apparent for the smaller datasets for these methods. Generally, these methods produce good results for running times around 2 minutes for the smaller images.

The running time for the other methods is more closely related to the sizes of the images. As can be seen from the table, the running time for the methods DAEU, MTAEU, and CNNAEU are much lower than for the comparison methods for the large image. The running time for the proposed method is around 2 minutes compared to 17 minutes for $\ell_{1/2}$-NMF and around 4 hours for the uDAS method. The running time for SHDP scales exponentially with image size, and therefore it was omitted from the comparison for the large image. The experiments were performed on a computer having an eight-core CPU and 64 GB of memory and a GPU having 11 GB of memory.

Chapter 5

Comprehensive Comparison of Autoencoder Methods for Unmixing

This chapter provides a comprehensive comparison of autoencoder-based methods for unmixing hyperspectral images. The comparison is performed on both real and synthetic datasets. The performance of 11 methods is evaluated and compared using four real datasets with accompanying reference endmembers and reference abundance maps. Furthermore, all methods are evaluated on four synthetic datasets having the same absolute ground truths but with different spectral variability. The results of both endmember extraction performance and abundance maps generation for all the methods are discussed, especially how they can be interpreted in light of the discussions in previous sections. Lastly, the computation costs of the methods are compared and discussed.

5.1 Experimental Results

This section provides an experimental foundation for chapter 2 and tries to determine experimentally what makes unmixing autoencoders perform well. The performance of 11 blind unmixing methods will be evaluated and compared using the Urban, Samson, Houston, and Apex datasets and accompanying reference endmembers and reference abundance maps. Additionally, all methods were evaluated on four synthetic datasets with increasing spectral variability.

Furthermore, all methods will be evaluated on four synthetic datasets having the same absolute ground truths but with different spectral variability. Also, ablation experiments will be performed on a very general and simple spectral unmixing autoencoder. Firstly, the ablation experiments aim to demonstrate what makes autoencoder-based methods powerful compared to traditional methods. Secondly, the effects of three different ways to enforce the ASC will be studied for two different activation functions for the encoder. Finally, the effects of non-linear vs linear decoders will be investigated.

The endmember extracted by methods will be evaluated using the mean spectral angle distance (mSAD), 1.6. Abundance maps generated by the methods will be compared to reference abundance maps using the RMSE measure given by 1.7 The results of both endmember extraction performance and abundance maps generation for all the methods will be discussed, especially how they can be interpreted in light of the discussions in previous sections. Lastly, the computation costs of the methods will be compared and discussed.

5.1.1 Methods compared in experiments

The methods compared in the experiments are listed in Table 5.1. The source code for methods number 4, 6, and 7 could not be obtained, so the authors implemented them. Method 12, sparsity regularized NMF, is a benchmark traditional method. The Endnet method was not published as an usable blind method as the generated abundance maps were of low quality due to particular implementation of a batch normalization layer and high ℓ_1-norm sparsity regularization. In our implementation, the batch normalization does not ruin the abundance maps, and neither does the chosen optimal strength of the ℓ_1 regularization. Therefore, the method produces usable abundance maps in our implementation. The method mDAE was initially implemented in Matlab, but we used the Tensorflow deep learning framework for our implementation. Methods number 1, 2, 3, and 4 are randomly initialized. All other methods are initialized using VCA.

5.1.2 Ablation Experiments

These experiments use a simple spectral unmixing autoencoder. Its architecture is shown in Fig. 5.1.

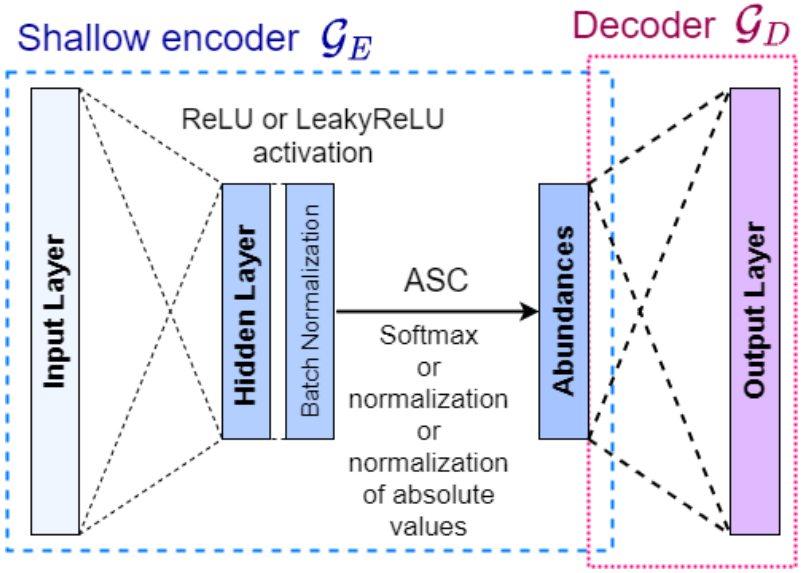


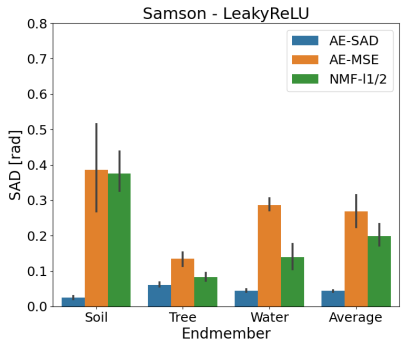
Figure 5.1: The architecture of the simple autoencoder used in the ablation experiments.

The encoder only has a single layer, and the activation can be either the ReLU or the Leaky ReLU activation with slope parameter 0.1 for the negative pre-activations. A batch normalization layer is used after the activation to speed up learning. The ASC

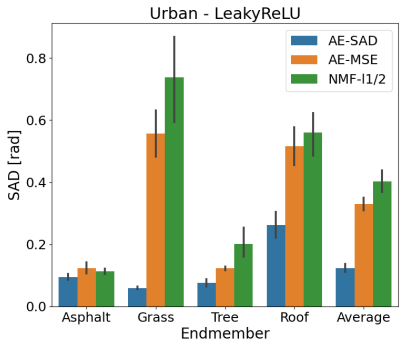
Table 5.1: The methods used in the experiments.

#	Name	Architecture	Loss	ASC	Regularization
1	DAEU [66]	Deep encoder	SAD	Uses (2.16)	
2	MTLAEU [67]	Branched deep encoder utilizing multitask learning with shared decoder	SAD	Scaled softmax	None
3	CNNAEU [64]	Fully CNN encoder and decoder	SAD	Scaled softmax	None
4	SIDAEU [81]	Shallow encoder	SID	Uses (2.16)	dynamical thresholding
5	OSPAEU [70]	Deep encoder	Hyper-Laplacian	Uses (2.18)	Orthogonal sparse prior
6	Endnet [77]	Single layer encoder with a custom matrix-vector product	SAD + small MSE term	Softmax	l_1 on abund. and l_2 on endm.
7	mDAU [12]	Marginalized denoising autoencoder in cascade with an unmixing autoencoder with tied weights	MSE	Uses augmented matrices	None
8	NLAEU [69]	Deep autoencoder nonlinear unmixing with a decoder having additionally two extra layers that model the interactions between the endmembers according to (1.4)	MSE	Uses (2.18) but no ANC	None
9	SNSA [129]	Stacked nonnegative sparse autoencoder with outlier detection prior to unmixing	MSE	Uses augmented matrices	minimum volume
10	uDAS [62]	An untied denoising autoencoder with sparsity	MSE	Uses augmented matrices	ℓ_{21} on endmembers
11	DAEN [83]	The method uses an variational autoencoder for spectral unmixing which uses the VAE indirectly to enforce the ASC constraint. The method optionally uses stacked autoencoders to prepare a outlier free training data set for the unmixing autoencoder	MSE	Uses VAE for ASC and ANC	Minimum volume
12	NMF-$\ell_{1/2}$ [39]	Sparsity constrained nonnegative matrix factorization ($\ell_{1/2}$-NMF) [39]	MSE	Uses augmented matrices	$\ell_{1/2}$ on abundances

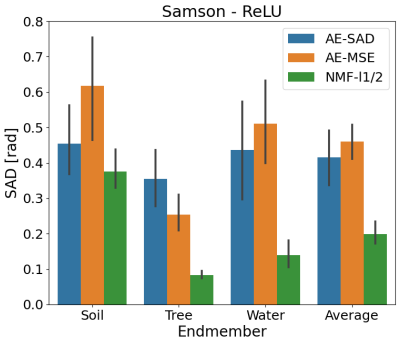
is enforced using one of the following: the softmax function, simple normalization according to (2.16) or normalization of absolute values according to (2.18). The loss function is either the SAD or MSE function.



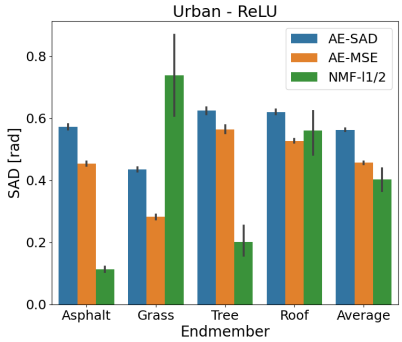
(A) Samson dataset



(B) Urban dataset



(C) Samson dataset



(D) Urban dataset

Figure 5.2: Comparison between a simple autoencoder using either the MSE or SAD function for the fidelity term, and the NMF-$\ell_{1/2}$ method. The autoencoder uses either the LReLU or ReLU activation. The black bars show the standard deviation. All experiments consisted of 25 runs.

Comparison with NMF-$\ell_{1/2}$ In this experiment, the simple autoencoder is compared with the NMF-$\ell_{1/2}$, a widely used NMF method that is sparsity regularized. The ASC will be enforced using the softmax function, and the ReLU activation function is used for the encoder. The loss function is either MSE or SAD. Fig. 5.2 shows the results of the experiment for the Urban and the Samson datasets.

Fig. 5.2 shows that a simple spectral unmixing autoencoder using the MSE function for the fidelity term and LReLU activation is only slightly better than the NMF-$\ell_{1/2}$ method for the Urban dataset and slightly worse for the Samson dataset. When the autoencoder uses the SAD loss function and the LReLU activation, the performance becomes significantly better than the performance of the NMF method. However, when the activation function is the ReLU, we see a much worse performance with the autoencoder.

This is because of ReLU units getting stuck when using the ReLU activation and batch normalization. The ReLU activation has a zero gradient for negative pre-activations, while the LReLU always has a nonzero gradient. This shows that care should be taken when using the ReLU activation as it can lead to bad performance.

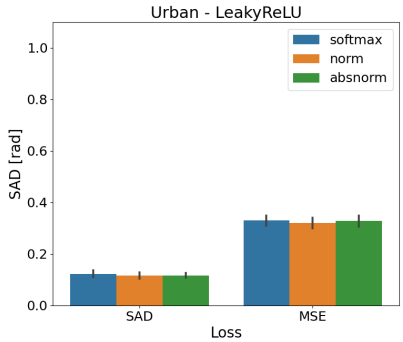
This experiment largely answers why autoencoder methods can achieve much better unmixing performance than methods using the MSE loss function. The scale-invariance of the SAD loss function makes the autoencoder able to handle the spectral variability inherent in most real HSIs.

Comparison between different ASC and activation combinations In this experiment, the effect of different ways to enforce the ASC will be compared for both MSE and SAD losses and using the ReLU and LReLU activations. Three different ways of enforcing the ASC will be tested:

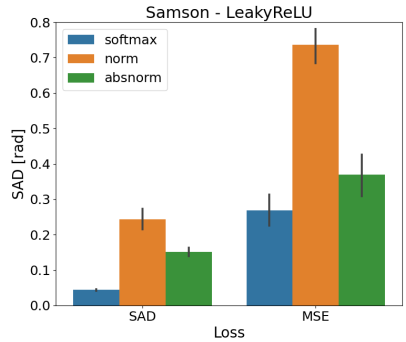
- Using the softmax function.
- Normalizing the activations according to (2.16).
- Normalizing the absolute value of the activations (absnorm) according to (2.18).

The results of this experiment are shown in Fig 5.3. The results are interesting but not surprising. For LReLU and the Urban dataset, the difference between the different ways of enforcing the ASC is negligible. However, for the Samson dataset with LReLU, softmax is by far the best option for both losses.

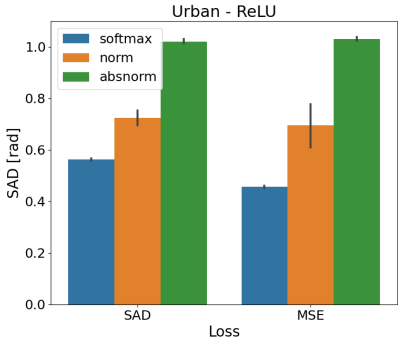
This difference is because LReLU allows negative activations, and for the Samson dataset, there seems to be a much more preference for negative activations than for the Urban dataset. Negative activations in the layer just before the ASC is enforced will negatively affect the simple normalization as it will break the constraint. It might



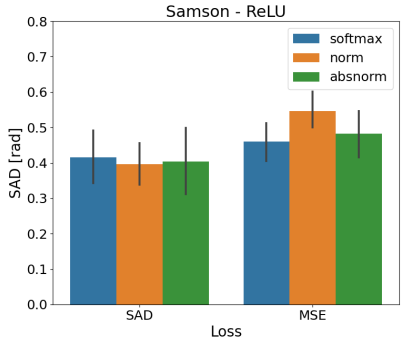
(A) Urban dataset with LReLU



(B) Samson dataset with LReLU



(C) Urban dataset with ReLU



(D) Samson dataset with ReLU

Figure 5.3: Comparison between three ways of enforcing the ASC constraint for both the MSE and SAD losses and using the LReLU or ReLU activations. The black bars show the standard deviation. All experiments consisted of 25 runs.

be argued that LReLU activation will lead to less sparse abundance maps than if ReLU activation combined with batch normalization is used. In deep encoders, a ReLU activation for the last layer might work well and give more sparse maps. The absnorm method is not as affected, but softmax does not care about the signs of the activations and will always return a vector that sums-to-one. It seems that the neural network can best adapt to the softmax method of enforcing the ASC.

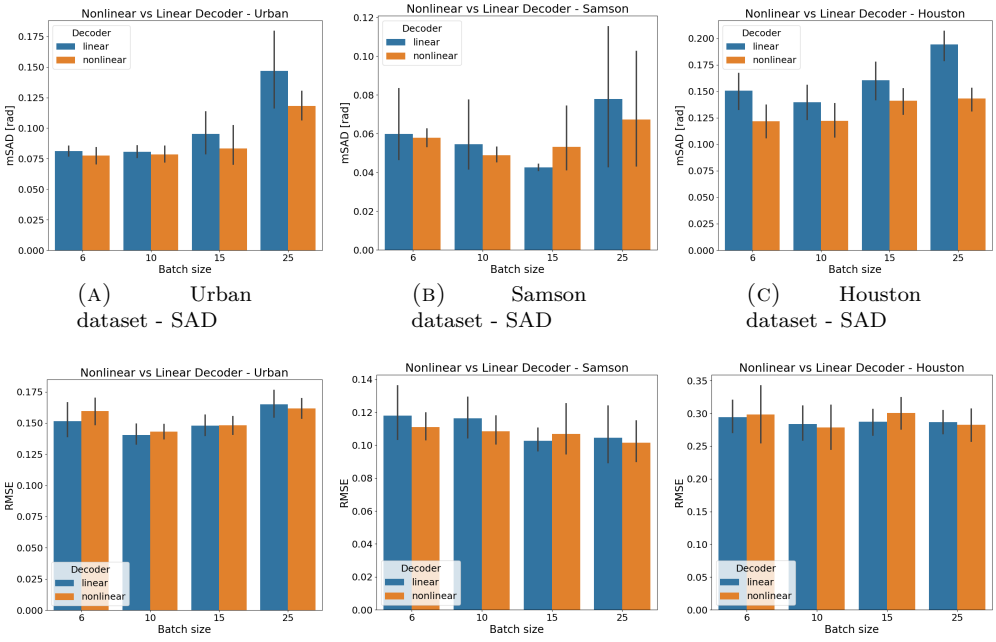
The ReLU part of this experiment is plagued by the vanishing gradient problem because of random initialization and is hard to interpret. This experiment shows that for some datasets, it can matter how the ASC is enforced. Using the softmax method seems always to work best and can be considered the safest option. By using a scaling factor for the activations entering the softmax function, the function can be soft thresholding and act very similarly to ℓ_1-sparsity regularization.

Linear and Nonlinear Decoders Until now, we have mainly discussed autoencoders having linear decoders where the endmembers are the weights of the decoder at the end of training. It is possible to have multi-layered and nonlinear decoders in spectral unmixing autoencoders. If the decoder has multiple layers, the endmembers cannot be extracted as the weights of any layers. However, the endmembers can be obtained from the decoder part by decoding one-hot vectors, i.e., the abundances of pure pixels.

In this final ablation experiment, we will compare two autoencoders with identical encoders, but one has a linear decoder and the other a nonlinear decoder. In the case of the nonlinear decoder, the endmembers are obtained by a prediction by the decoder part on the identity matrix (abundances of pure pixels). The nonlinear decoder has three layers: one with $4\times R$ units, and two layers with B units each, where R is the number of estimated endmembers and B is the number of bands in the image. This particular decoder does not correspond to any well known nonlinear mixing model and cannot even be said to be post-nonlinear. However, because of the great flexibility of deep learning models, it would be easy to implement most commonly used nonlinear mixing models. Especially models of the post-nonlinear variety as described by 1.4. For examples on nonlinear unmixing models implemented via autoencoders, see [69,93,96]

Two datasets are used, the Urban and Samson datasets and the LReLU activation was used for the encoder. Four different batch sizes are used: 6, 10, 15, and 25. Fig. 5.4 shows the results of the experiment. The figures in the top row are the mSAD of extracted endmembers, and the figures in the bottom row show the RMSE for the abundance maps.

What first stands out when looking at Fig. 5.4 is that the batch size is a critical hyperparameter, especially for the endmembers. The best SAD performance on the Urban and Houston images is for small batch sizes, while for Samson, it is best to use moderately large batch size. It can be seen that for the Urban image, having more parameters in the decoder is beneficial for the mSAD scores of extracted endmembers.



(A) Urban dataset - SAD (B) Samson dataset - SAD (C) Houston dataset - SAD

(D) Urban dataset - RMSE (E) Samson dataset - RMSE (F) Houston dataset - RMSE

Figure 5.4: Comparison between having a linear or nonlinear decoder. The encoder uses the LReLU activation. Four different batch sizes are tested. The graphs in the top row show the average SAD for extracted endmembers while the bottom row shows the RMSE of the abundances. The black lines show the standard deviation. All experiments consisted of 25 runs.

The case of the Samson image is not as clear and depends on the batch size. The results for the Houston image indicate clearly that a nonlinear decoder results in better mSAD scores, indicating that this image has more nonlinear mixing than the other images. Regarding the quality of the abundance maps, the experiment is not conclusive. However, it is interesting to see that a nonlinear decoder that does not correspond to the LMM can benefit endmember extraction while the quality of abundance maps does not change much.

Summary The results of the ablation experiments can now be summarized:

- Autoencoder methods should use a scale-invariant loss such as SAD, and the use of MSE should be avoided unless the methods specifically handle spectral variability through an extended LMM.
- Using the LReLU activation in combination with softmax to enforce the ASC works well, but abundance maps might be less sparse than if ReLU is used. Care should be taken when using the ReLU activation to prevent units getting stuck.
- Batch size affects the performance of unmixing autoencoders and is dataset dependent.
- Using a nonlinear decoder with more parameters can benefit the quality of extracted endmembers, especially for images with nonlinear mixing. The effect on the quality of the abundance maps is not very significant.

5.1.3 Experiments with real HSIs

Endmember Extraction

When evaluating the endmember extraction performance of the methods, three qualities will be in focus:

- The average SAD from the reference endmembers.
- The variance of the average SAD score. A good method should have a low variance.
- The consistency of the method is studied.

Consistency is not quite the same thing as variance. Here we are looking at if the method finds the same endmembers every time or most of the time. A method having bad consistency might, e.g., be confusing one endmember for a variant of another for some runs. Plotting all extracted endmembers for every run and showing these together in a single figure is an excellent way to evaluate the methods. The Urban, Houston, and Apex datasets were used to evaluate the endmember extraction performance of the methods.

Urban dataset In the first experiment, we use the Urban dataset, extract the first four, then five, and finally six endmembers, and compare the extracted endmembers to reference endmembers. As the number of estimated endmembers increases, it becomes harder to get good solutions, and it is interesting to see how the solutions for the initial endmembers behave as new endmembers, in addition, are estimated. Figures 5.5, 5.6, 5.7 show extracted endmembers for all the methods for the Urban dataset with four, five, and six reference endmembers, respectively. The reference endmembers are shown with red colour. Tables 5.2, 5.3, 5.4 tabulate the average SAD from reference endmembers in radians along with the standard deviation for the Urban dataset with four, five, and six reference endmembers, respectively.

Table 5.2: The mean SAD($\times 10^{-2}$) from reference endmembers in radians along with the standard deviation for all methods for the Urban dataset with four reference endmembers. Best results are in bold.

Endmember Method	Asphalt	Grass	Tree	Roof	Average
CNNAEU	**5.75±0.59**	**3.66±0.48**	**3.21±0.4**	**3.32±0.67**	**3.98±0.31**
MTAEU	8.43±0.47	4.21±0.37	5.4±0.4	4.15±0.46	5.6±0.2
DAEU	7.2±2.4	7.7±2.9	7.07±2.97	21.55±8.62	10.88±2.39
SIDAEU	10.39±3.86	9.95±6.63	6.4±2.3	21.5±12.1	12.06±3.23
OSPAEU	12.9±1.3	9.0±1.9	13.5±6.0	20.1±27.2	13.86±6.35
Endnet	6.18±1.18	5.38±1.69	3.63±0.45	5.08±1.61	5.1±0.5
mDAE	16.26±3.78	81.2±38.8	48.5±41.0	49.5±14.0	48.9±7.5
NLAEU	20.2±11.3	43.0±31.1	93.4±39.1	22.7±10.5	44.9±3.9
SNSA	28.58±2.14	120.2±15.7	7.88±0.14	12.0±5.8	42.2±2.8
uDAS	21.04±1.98	107.2±20.0	14.0±2.5	26.5±4.3	42.2±6.3

Table 5.3: The mean SAD from reference endmembers in radians along with the standard deviation for all methods for the Urban dataset with five reference endmembers. Best results are in bold.

Endmember Method	Asphalt	Grass	Tree	Roof	Soil	Average
CNNAEU	3.79±0.31	6.11±0.97	13.2±0.9	**4.04±0.69**	10.75±0.98	**7.58±0.34**
MTAEU	12.11±23.63	5.65±3.54	11.32±1.68	6.61±0.96	16.52±4.67	10.44±4.82
DAEU	11.64±6.03	20.05±7.09	14.0±3.4	41.71±8.21	31.6±18.0	23.8±3.3
SIDAEU	23.8±22.7	20.0±6.3	9.9±2.1	15.1±11.2	43.4±27.2	22.5±3.0
OSPAEU	36.0±33.2	7.72±1.38	**8.94±3.06**	30.1±28.7	**5.07±1.23**	17.6±6.1
Endnet	**3.68±0.56**	**4.06±1.17**	11.71±1.12	13.13±4.63	15.1±3.3	9.53±0.87
mDAE	22.2±6.2	94.2±32.5	77.8±40.1	53.4±15.8	67.5±22.7	63.0±7.8
NLAEU	21.5±14.7	86.2±52.6	33.6±21.9	12.12±5.13	46.9±35.3	40.1±4.7
SNSA	79.1±31.3	77.6±16.6	15.1±0.8	21.6±11.6	44.0±14.8	47.5±5.1
uDAS	16.7±4.0	59.2±49.7	17.0±4.7	21.3±4.8	78.2±42.4	38.5±4.1

When estimating four endmembers, CNNAEU, MTAEU, and Endnet show good performance, low variance, and good consistency. The DAEN method has low variance

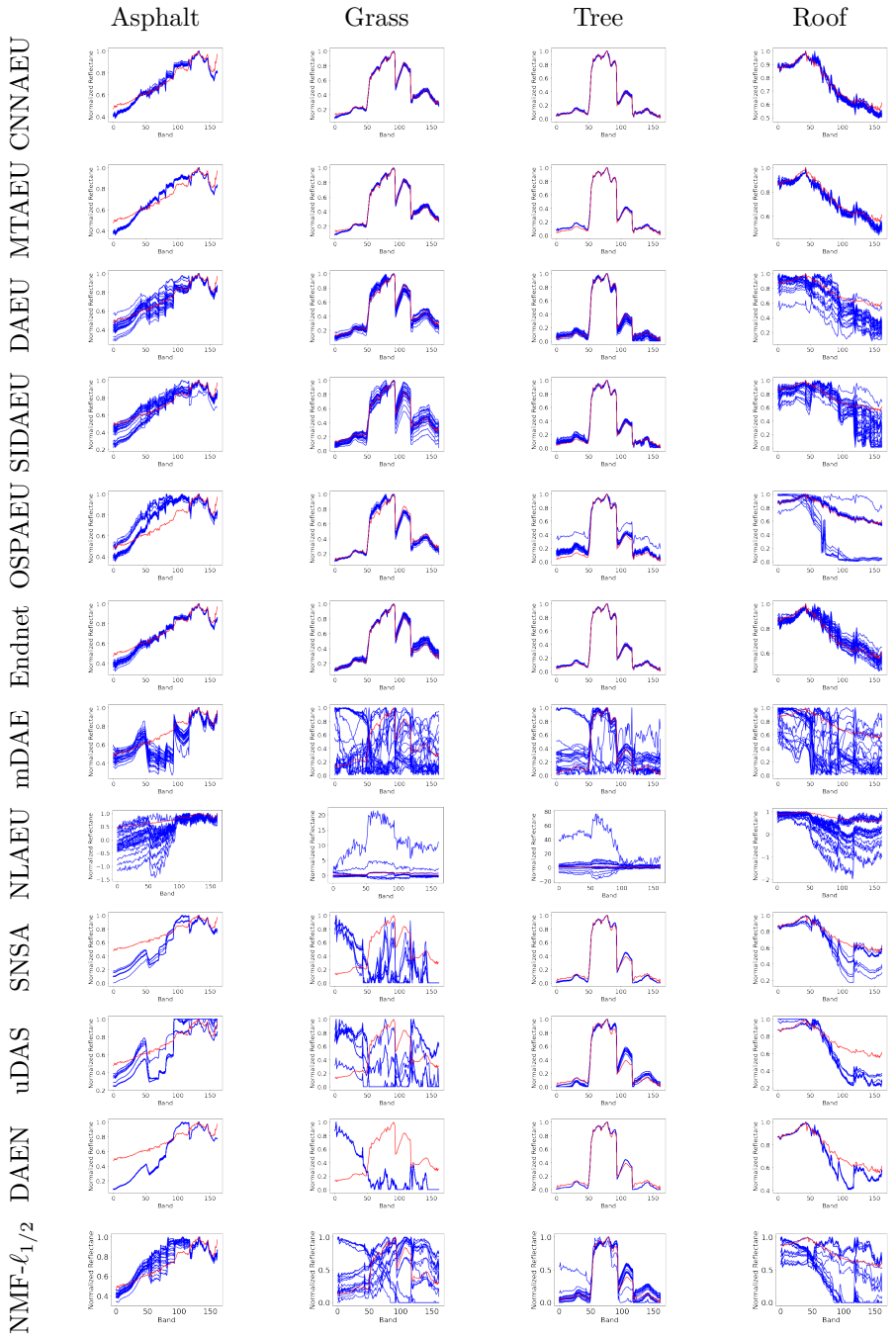


Figure 5.5: Plots of all extracted endmembers by all methods (blue curves) and the reference endmembers (red) for the Urban dataset with four reference endmembers.

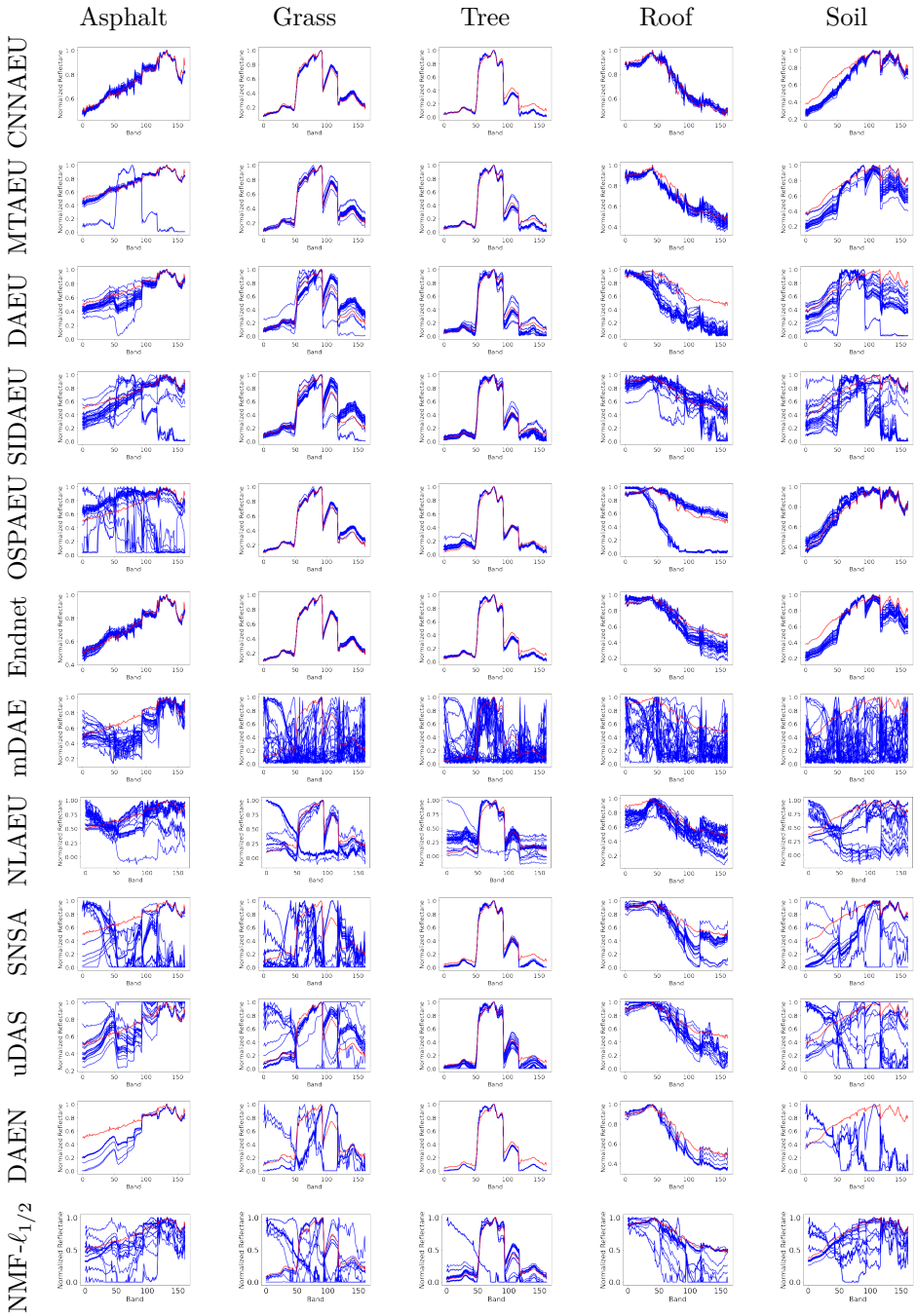


Figure 5.6: Plots of all extracted endmembers by all methods (blue curves) and the reference endmembers (red) for the Urban dataset with five reference endmembers.

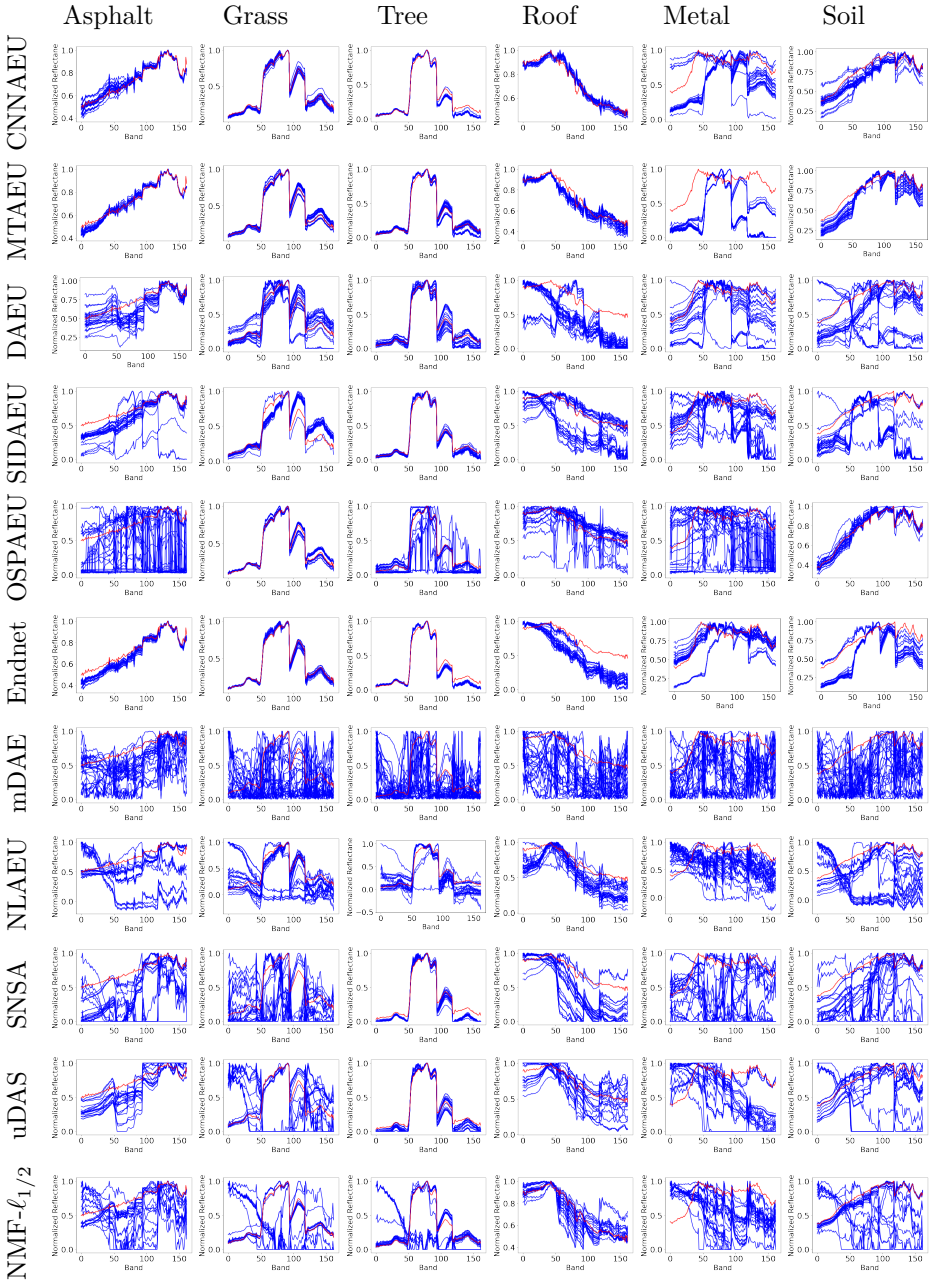


Figure 5.7: Plots of all extracted endmembers by all methods (blue curves) and the reference endmembers (red) for the Urban dataset with six reference endmembers.

Table 5.4: The mean SAD from reference endmembers in radians along with the standard deviation for all methods for the Urban dataset with six reference endmembers. Best results are in bold.

Endmember Method	Asphalt	Grass	Tree	Roof	Metal	Soil	Average
CNNAEU	**4.3±1.0**	**4.9±2.5**	13.39±1.66	**4.3±0.8**	36.3±11.3	10.3±3.0	**12.3±1.7**
MTAEU	4.5±0.7	7.5±2.1	**7.7±4.0**	6.1±0.8	7.1±18.1	12.1±4.1	18.1±3.0
DAEU	16.3±7.0	21.6±12.1	13.0±4.6	44.0±6.9	38.0±30.5	45.3±31.9	29.7±4.2
SIDAEU	21.9±20.7	23.7±3.2	8.74±1.48	28.0±17.6	37.5±24.7	58.0±25.0	29.6±2.9
OSPAEU	66.4±29.8	6.83±1.47	26.8±26.9	14.4±13.8	50.0±25.7	**4.5±1.5**	28.2±7.1
Endnet	5.81±1.04	5.49±2.15	12.5±1.1	31.1±6.1	**15.1±9.3**	24.3±6.8	15.7±1.3
mDAE	31.4±14.8	79.7±32.1	90.3±38.0	49.9±17.5	52.7±20.0	55.8±22.9	60.0±6.0
NLAEU	31.4±29.0	51.0±48.4	28.8±23.0	14.8±4.6	38.1±14.1	51.6±39.6	35.9±4.4
SNSA	52.4±22.1	76.3±26.9	16.3±2.2	46.1±15.4	67.1±30.8	53.1±29.8	51.9±3.8
uDAS	24.3±9.0	89.8±52.3	14.1±5.8	22.9±10.5	56.1±20.6	37.0±40.9	40.7±5.8

and good consistency but a bad SAD score. The mDAE method, which is the only method with tied weights, is very unstable, and this instability can be attributed to tying the weights and spectral variability in the image. The method NLAEU has highly negative endmembers as it lacks a nonnegativity constraint. This makes the method essentially unusable when not initialized with "good" endmembers.

The methods uDAS, SNSA, and DAEN interestingly all yield similar endmembers, and they are the only methods implemented in Matlab and not using an established deep learning framework such as Tensorflow or Pytorch. The DAEU and SIDAEU methods have high variance, medium consistency, and not very good SAD scores. The OSPAEU technique has bad consistency and high variance. Finally, the "Roof" endmember oscillates between three different solutions indicating some instability in the training process.

As the number of estimated endmembers are increased to five, the average SAD scores of all methods increase and the consistency of many of the methods decreases significantly, with endmembers that previously had good consistency become unstable. Again, CNNAEU, MTAEU, and Endnet show the best performance and consistency, with CNNAEU having the lowest mSAD and variance. MTAEU has one inconsistent run where it estimates the Asphalt endmember as a "Tree" endmember variant, resulting in worse mSAD and higher variance.

The OSPAEU technique estimates the new "Soil" endmember quite well, but the "Asphalt" endmember, which previously, when estimating four endmembers was good, becomes unstable. Interestingly, the NLAEU method, which previously yielded highly negative endmembers, estimates them consistently mostly nonnegative. Still, the consistency is not good. DAEN, uDAS, and SNSA seem to have trouble estimating the new "Soil" endmembers.

When estimating six endmembers, the trend of increasing average SAD and decreasing

consistency continues. The best performing methods are again CNNAEU, MTAEU, and Endnet. CNNAEU and MTAEU have greater difficulty estimating the "Metal" endmember than the Endnet method, but they retain excellent consistency and good SAD scores for the five other endmembers. Both methods are spectral-spatial methods. For Endnet, the cost of a good "Metal" endmember is a poorer solution for the "Soil" endmember. All the other methods show bad consistency for most endmembers other than "Tree", which is the best-represented endmember in the Urban scene.

Interestingly, the only method to consistently estimate the "Metal" endmember well is the Endnet method. This method uses a highly customized hidden layer in the encoder where the standard dot product used to calculate the layer's activations from its weight matrix and inputs is replaced with a custom and more "spectrally discriminating" dot product based on the SAD similarity measure. For some reason, the DAEN method sometimes took too long to extract six endmembers for this dataset, so it had to be omitted from this experiment.

Table 5.5: The mean SAD ($\times 10^{-2}$) from reference endmembers in radians along with the standard deviation for all methods for the Houston dataset with four reference endmembers. Best results are in bold.

Method	parking_lot1	parking_lot2	running_track	grass_healthy	Average
CNNAEU	3.13±0.51	7.53±0.96	**7.91±7.21**	4.11±0.91	**5.66±2.14**
MTAEU	4.27±0.92	7.86±1.30	28.9±9.1	6.42±1.06	11.87±2.31
DAEU	12.63±3.09	28.0±16.2	11.4±4.0	9.37±8.42	15.3±3.6
SIDAEU	9.65±2.31	15.6±4.7	25.3±13.8	6.91±5.59	14.4±3.6
OSPAEU	**1.27±0.31**	10.58±0.26	13.4±2.6	10.28±1.14	8.88±0.63
Endnet	8.00±0.75	**6.35±0.51**	23.4±1.2	**1.65±0.32**	9.85±0.36
mDAE	33.5±24.8	55.4±26.8	27.7±8.3	24.4±8.21	35.2±6.1
NLAEU	24.2±18.2	66.8±16.3	21.6±0.5	11.6±1.0	31.1±0.8
SNSA	9.71±9.72	82.7±3.7	49.6±20.3	24.9±1.8	41.7±5.8
uDAS	11.3±11.2	30.4±15.2	15.5±5.4	10.8±5.7	0.17±0.0396

Houston dataset The extracted endmembers by all methods for the Houston dataset are shown in Fig. 5.8, and the SAD scores for individual endmembers along with the average SAD is tabulated in Table 5.5. The three best-performing methods for this dataset are CNNAEU, OSPAEU, and Endnet. Endnet has excellent consistency, but its solution for the "running_track" endmember is different from the reference. CNNAEU also has excellent consistency, with only one run out of 25 coming up with a different solution for the "running_track" endmember, and it also achieves the lowest average SAD score. OSPAEU has some problem with the "running_track" endmember and oscillates between two solutions, one of them being the reference endmember. The method achieves the second-lowest average SAD score for the dataset.

MTAEU also has trouble with the "running_track" endmember, showing similar behaviour as OSPAEU. The DAEU and SIDAEU methods have high variances despite having reasonably good consistency. mDAE does better on this dataset than it did on Urban but is still unstable because of the tied weights. The methods SNSA, uDAS and

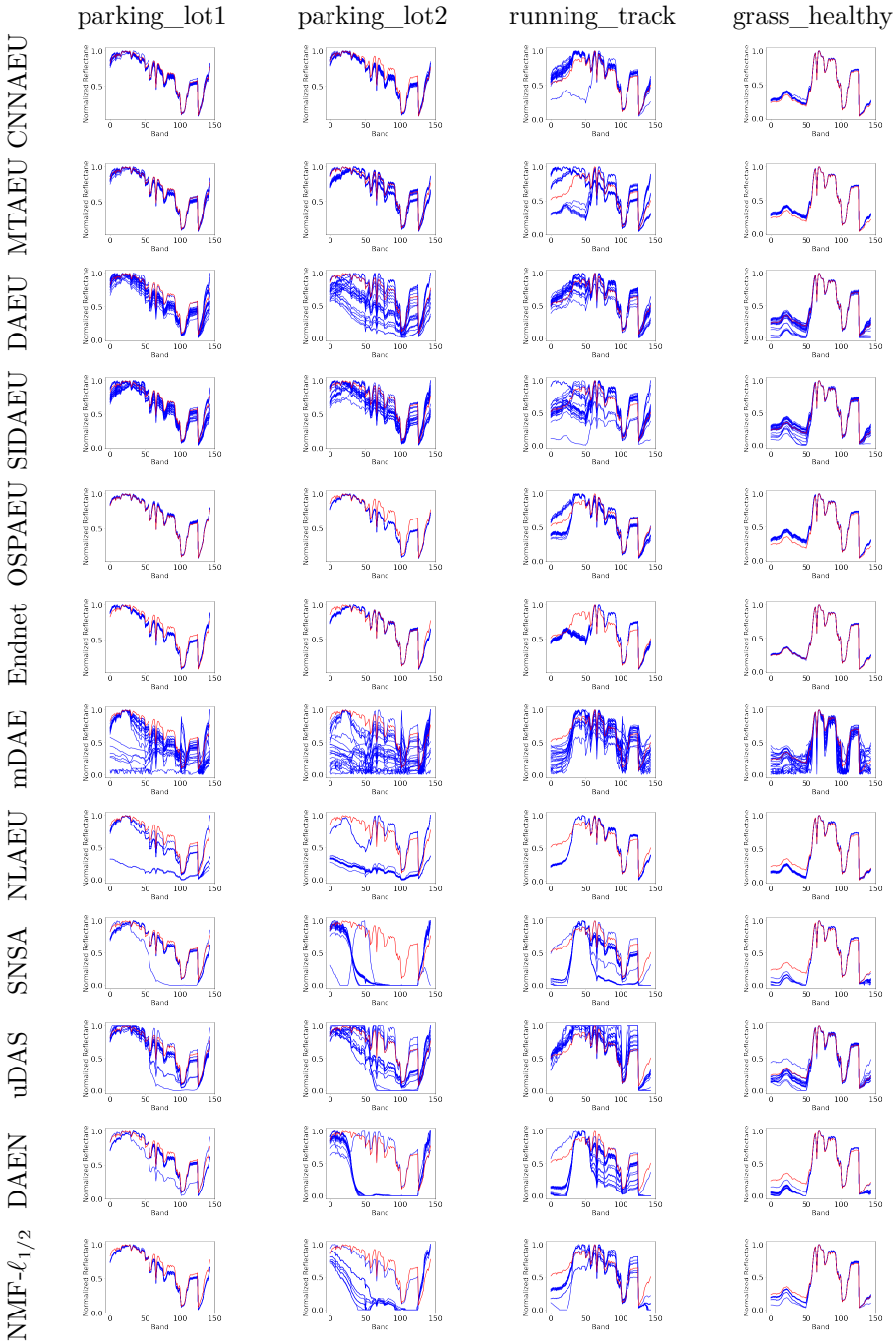


Figure 5.8: Plots of all extracted endmembers by all methods (blue curves) and the reference endmembers (red) for the Houston dataset with four reference endmembers.

DAEN all have trouble with the parking_lot2 endmember, which is highly correlated to the parking_lot1 endmember. Also, only uDAS is extracting the "running_track" endmember similar to the reference. NLAEU has some trouble with the parking_lot endmembers, and the consistency is not good with these endmembers. Its solution for the "running_track" endmember has a good consistency, but it is not the reference endmember.

Overall, the methods are doing better on the average on this dataset than on the Urban dataset. The difference in performance between methods that use scale-invariant fidelity terms and those that do not is less than for the Urban dataset, indicating that the Houston dataset has less spectral variability concerning scaling of spectra. The common ambiguity in the solutions for the "running_track" endmember is probably because it is the most under-represented endmember.

Apex dataset The Apex dataset is the final dataset for the evaluation of blind unmixing performance. Fig. 5.9 shows all extracted endmembers for all the methods while Table 5.6 tabulates the average SAD scores for individual endmembers along with the mSAD score. This dataset has very distinct and easily identifiable endmembers. The water endmember in this dataset can be challenging because the water spectra have a tiny scale compared to the other endmembers. Methods using scale sensitive similarity measures such as MSE can have difficulties with extracting such endmembers. The methods can often lower the objective function by extracting a variant of a well-represented endmember having a much larger scale instead of the sought after endmember. An endmember having a tiny scale can also manifest as a poorly learned endmember having irregularities and generally being badly formed.

Table 5.6: The mean SAD ($\times 10^{-2}$) from reference endmembers in radians along with the standard deviation for all methods for the Apex dataset with four reference endmembers. Best results are in bold.

Method	Road	Water	Roof	Grass	Average
CNNAEU	**5.87±2.00**	**4.17±0.37**	12.33±0.61	6.21±0.38	**7.14±0.46**
MTAEU	11.3±4.1	13.7±1.7	54.6±25.1	9.92±0.77	22.4±7.3
DAEU	36.4±11.4	10.6±2.5	23.2±17.7	11.9±3.8	20.5±3.5
SIDAEU	34.4±4.5	17.0±4.4	12.84±2.29	**5.61±1.67**	17.5±2.3
OSPAEU	7.5±0.9	6.62±1.44	16.5±25.2	13.2±2.5	10.95±6.07
Endnet	9.0±1.0	109.9±0.4	**5.15±0.68**	9.35±0.79	33.3±0.4
mDAE	48.2±17.7	105.7±18.4	23.0±8.8	49.6±7.7	56.6±6.3
NLAEU	39.7±9.0	71.8±16.3	14.0±2.1	38.2±6.7	40.9±5.7
SNSA	63.2±18.3	56.7±9.3	9.46±6.16	6.07±0.24	33.9±5.4
uDAS	30.8±10.4	58.3±15.9	12.6±5.0	10.0±4.8	27.9±5.9

Looking at Fig. 5.9, we see that all methods utilizing SAD as the similarity measure in the loss function have no trouble extracting the "Water" endmember, and it is very close

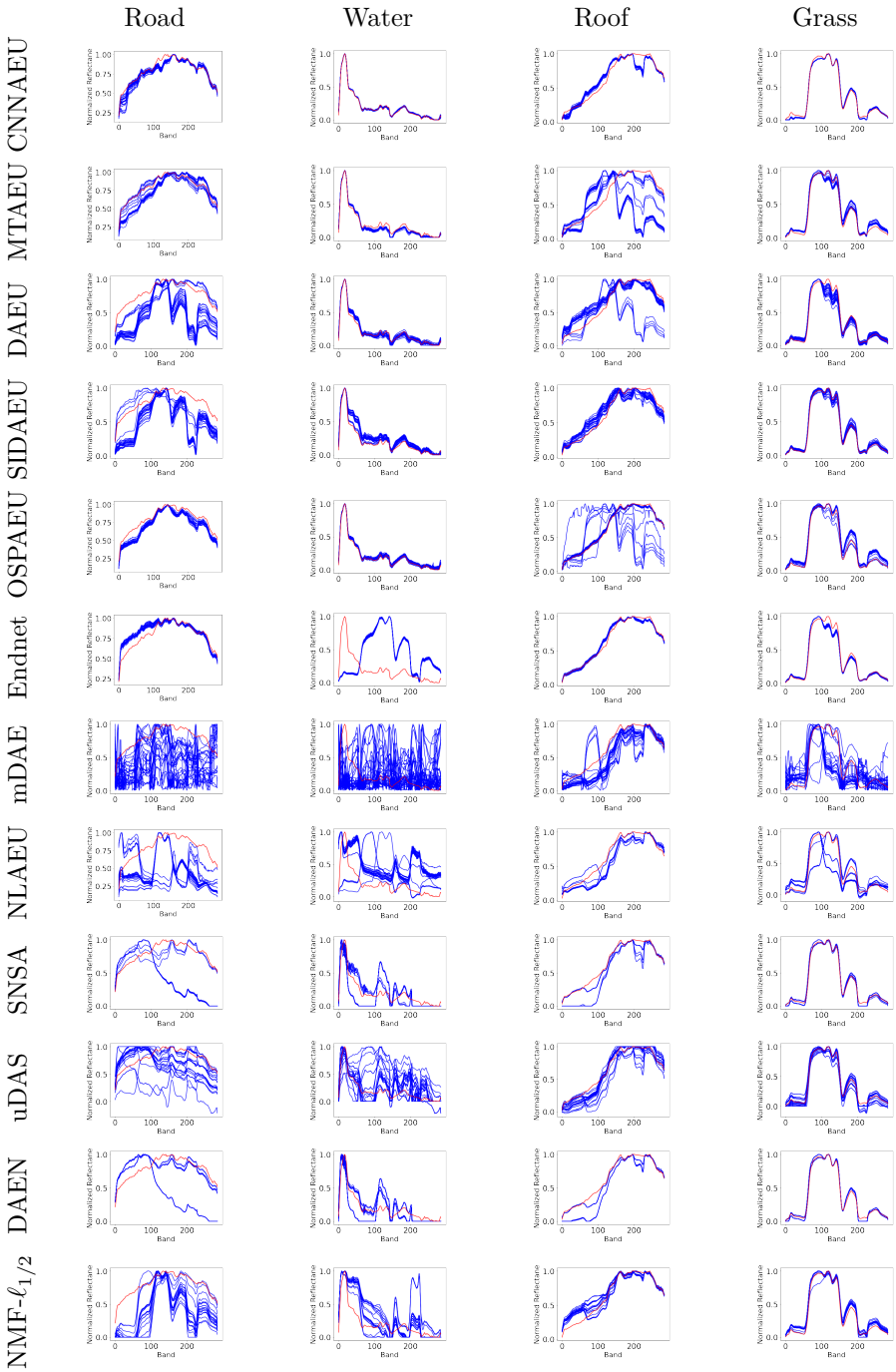


Figure 5.9: Plots of all extracted endmembers by all methods (blue curves) and the reference endmembers (red) for the Apex dataset with four reference endmembers.

to the reference. The water spectra in the dataset have minimal spectral variability, which explains the excellent match. CNNAEU has excellent consistency and only a slight variance for this dataset. OSPAEU, which does use a scale sensitive fidelity term, the hyper-Laplacian loss, does manage to extract the "Water" endmember very well. The loss is essentially a p-norm of the difference between input and reconstruction raised to the power p and having $p = 0.7$, which can explain why the lower scale of the endmember is not as problematic as it is for the MSE loss.

The Endnet method fails consistently to extract the "Water" endmember and extracts some vegetation endmember instead. This can be explained by the MSE term in the loss of this method. Despite this, the method has a good consistency and low variance. The MTAEU and DAEU methods fail to consistently extract the correct form of the "Roof" endmember and extract some vegetation endmember instead. It is hard to attribute this to some implementation details without doing some experimentation. However, the methods that have bad consistency for the "Roof" endmember all have deep encoders, i.e., MTAEU, DAEU, and OSPAEU. NLAEU, SNSA, uDAS, and DAEN all have trouble with the "Road" and the "Water" endmember.

Abundance maps

The quality of generated abundance maps by blind unmixing methods depends on the quality of the extracted endmembers. The discussion of abundance maps will be limited to the Urban dataset. Fig. 5.10 shows the generated abundance maps by all methods for the best mSAD score. Table 5.7 tabulates the RMSE scores for individual endmembers and the average of all maps.

Table 5.7: The mean RMSE between generated abundance maps and reference maps along with the standard deviation for the Urban dataset with four reference endmembers. Best results are in bold.

Method	Asphalt	Grass	Tree	Roof	Average
CNNAEU	25.7±0.8	29.4±1.2	20.8±1.7	16.8±1.9	23.7±1.0
MTAEU	15.2±0.3	**15.0±0.5**	**8.2±0.5**	**8.9±0.4**	**12.3±0.3**
DAEU	16.9±4.8	22.9±5.6	18.6±6.9	13.14±4.1	18.6±3.8
SIDAEU	17.2±5.5	19.7±5.2	16.3±3.8	16.2±6.2	17.7±4.2
OSPAEU	26.6±4.6	34.6±3.0	30.9±4.5	19.4±4.7	28.7±2.0
Endnet	**14.21±2.63**	17.4±1.5	8.96±1.09	11.02±2.38	13.4±1.3
mDAE	34.3±3.9	32.7±4.1	36.2±3.3	29.5±7.4	33.5±2.6
NLAEU	27.4±1.1	34.5±3.4	37.4±7.5	17.5±3.1	30.5±1.3
SNSA	33.6±1.0	41.1±0.2	35.4±0.4	20.7±0.2	33.5±0.4
uDAS	32.7±1.0	43.4±1.8	31.4±2.0	20.4±0.9	33.0±0.3

The method that achieves the lowest RMSE score is the MTAEU method. CNNAEU, despite having a lower mSAD score, does not achieve a good RMSE score. Fig. 5.10 shows that the abundance maps for CNNAEU are intense and sparse. This is a

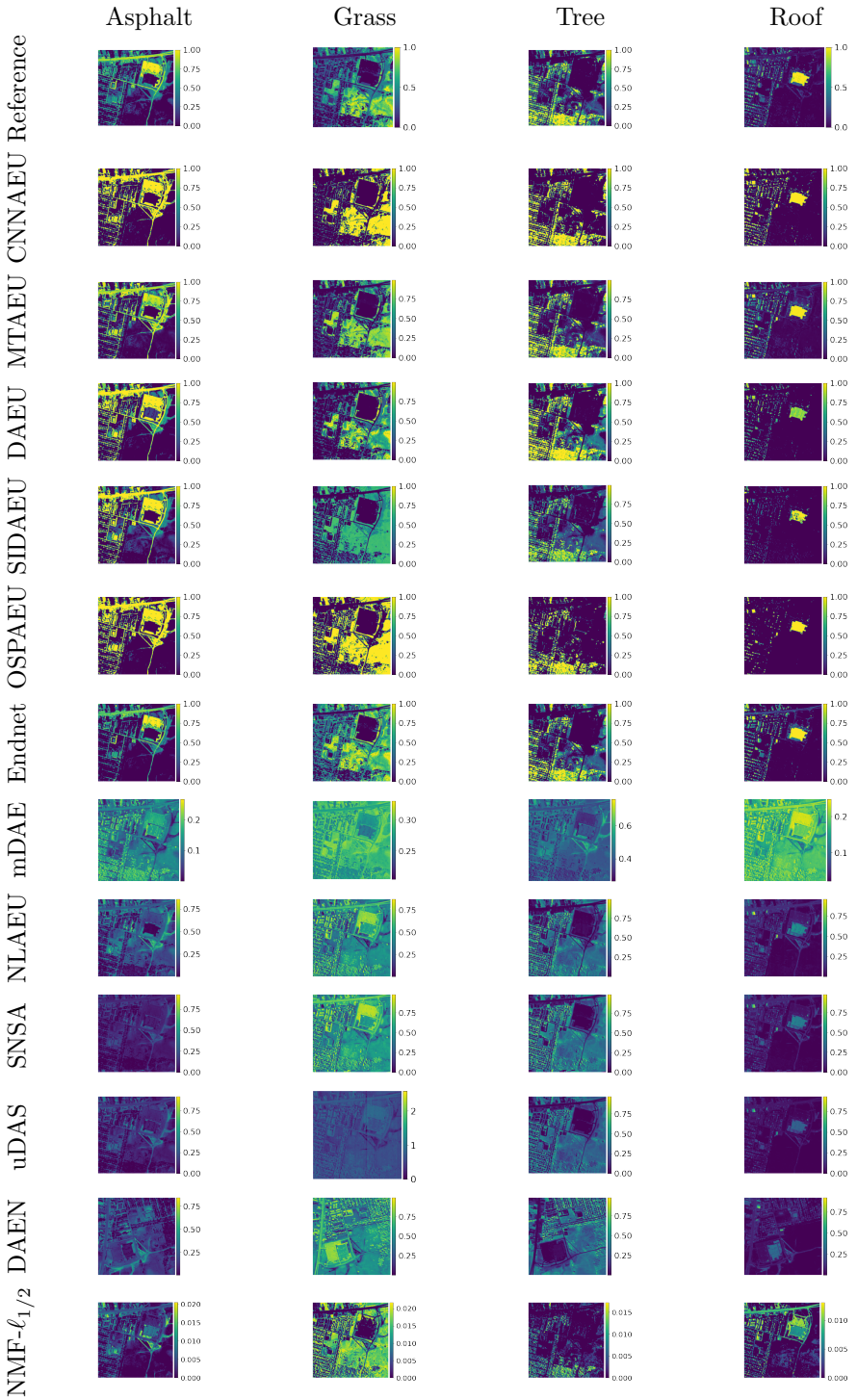


Figure 5.10: The abundance maps for the run with the best mSAD score for all methods for the Urban dataset. The reference abundance maps are in the top row.

consequence of using the softmax function to enforce the ASC and using a sizeable spatial filter for the decoder convolutional layer, causing high values of the feature maps entering the softmax function, which acts like ℓ_1-sparsity regularization. This could have been fixed by dividing the pre-activation with a number that depends on the filter size of the decoder.

The OSPAEU method also produces very intense and binary looking abundance maps. This results likely from the orthogonality prior of the abundance maps. The Endnet method achieves the second-lowest RMSE score and produces good abundance maps. This method employs forced sparsity by applying a "top_k" function to the abundances, selecting the k highest abundances and setting all the others to zero. A value of $k = 2$ was used in the experiments.

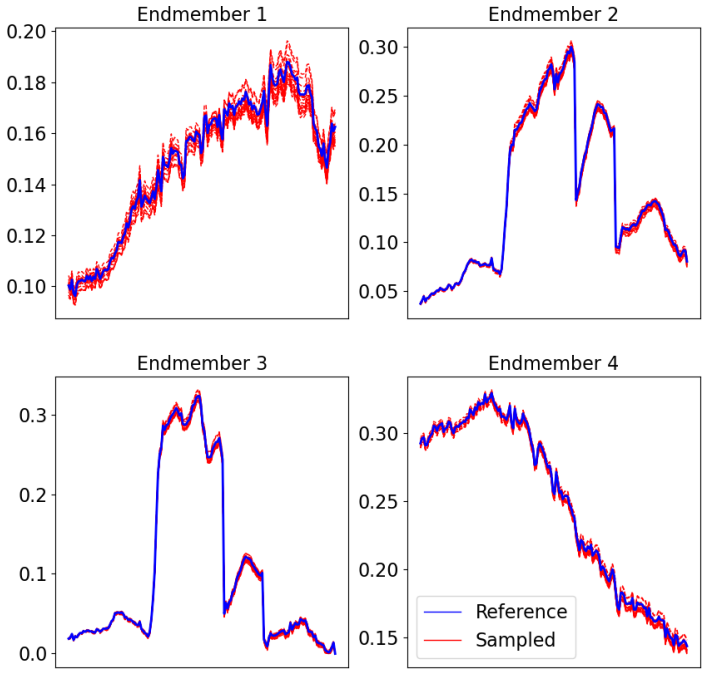
It must be noted that the authors of the Endnet method did not intend it to be used for generating abundance maps because of how they implemented a custom batch normalization layer that resulted in very poor maps. However, in our implementation of their method, the abundance maps look good and are perfectly usable. The abundance maps of the other methods are not good because the extracted endmembers by these methods are not good.

5.1.4 Synthetic datasets with varying spectral variability

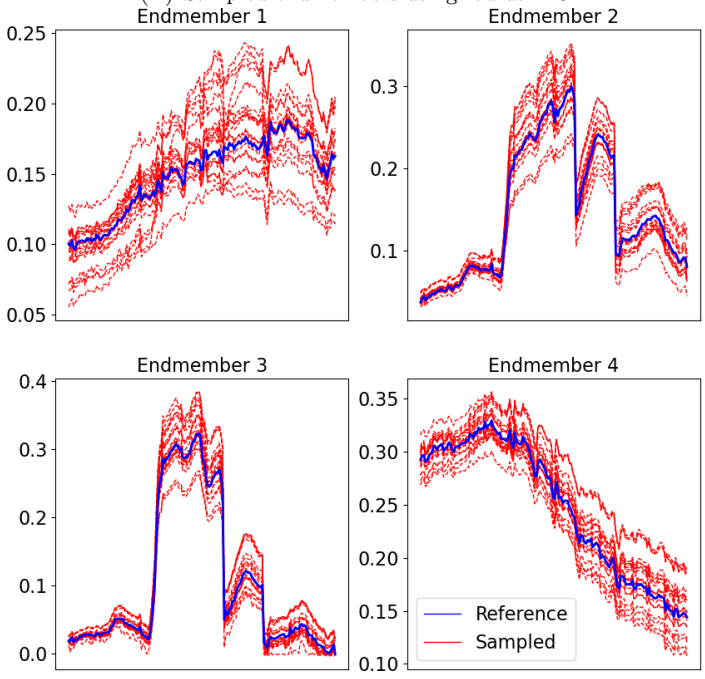
For this experiment, a variational autoencoder was used to learn the Urban image and the reference endmembers (4 endmembers) were encoded into codes in the 2D latent space of the encoder. By sampling within a circle in the latent space centred on the codes of the reference endmembers, new variations of the endmembers can be generated using the generator part, and the radius of the sampling circle controls the spectral variability of the sampled endmembers. A 100 by 100 pixel crop of the Urban dataset reference abundance maps was then used to provide the ground truth abundance fractions for every pixel, and new spectra generated by using sampled endmembers for every pixel in the abundance maps.

Four such datasets were generated with sampling radii of 0, 10, 20 and 40. Fig. 5.11 shows examples of sampled endmembers for two different radii. Every method was run 25 times for each of these synthetic datasets, and a bar plot of the average SAD for each dataset is shown in Fig. 5.12.

Fig. 5.12 is very informative. It shows clearly the effect of spectral variability on the performance of methods using the MSE objective function. The six rightmost methods all use the MSE fidelity term, and they all achieve very good mSAD for zero spectral variability. Even the mDAE method that performs very poorly on real datasets achieves good mSAD for zero spectral variability. It is striking to see how similar the performance of the NAE-VCA, SNSA, uDAS, and DAEN is to NMF-$\ell_{1/2}$ on these datasets. As the spectral variability increases, the non-scale-invariant methods perform increasingly worse.



(A) Sampled endmembers using radius = 5



(B) Sampled endmembers using radius = 40

Figure 5.11: Sampled endmembers for the synthetic experiment based on the Urban image for two different sampling radii.

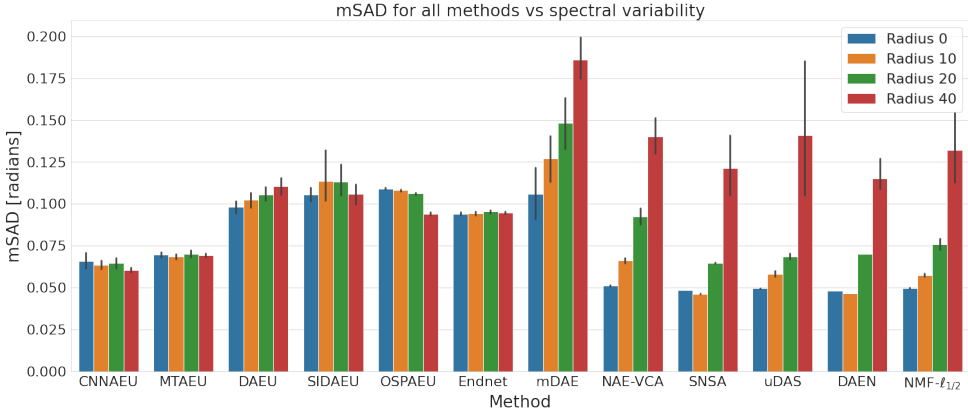


Figure 5.12: A bar plot showing the average SAD for the four synthetic datasets. The spectral variability increases with increasing sampling radius. The six rightmost methods all use the MSE objective function. The black vertical lines show the standard deviation. All experiments consisted of 25 runs.

The methods using scale-invariant fidelity terms exhibit excellent robustness against spectral variability. Only DAEU performs increasingly worse with increasing spectral variability. The spectral-spatial methods CNNAEU and MTAEU show the best overall performance. The results of this experiment indicate that synthetic datasets with low or zero spectral variability is not a good benchmark for autoencoder methods in general. Low spectral variability results in bias towards non-scale-invariant fidelity terms on such datasets that would not be reflected on real datasets.

5.1.5 Robustness to noise

A blind unmixing experiment was performed using the Samson dataset to investigate the methods' robustness to noise. The dataset was corrupted with noise to obtain four versions having signal-to-noise-ratio (SNR) of 10, 20, 30, and 40 dB. Fig. 5.13 shows a bar chart of the mSAD score of all the methods for the Samson dataset for the four different SNR levels and the original uncorrupted dataset. The figure shows that most methods are relatively robust to noise. MTAEU and OSPAEU seem largely unaffected by the noise for this dataset. Endnet shows good performance and low variance for all SNR levels but is relatively largely affected by SNR = 10 dB. Paradoxically, SNSA and DAEN perform best at SNR = 10 dB. NLAEU is heavily affected by noise, and SIDAEU does not handle SNR = 10 dB well. CNNAEU performs best under moderate noise (SNR=30) corruption and also has low variance for this SNR. On the original dataset, Endnet and MTAEU show the best performance and least variance. The NMF method performs best at SNR = 20 and 30 dB.

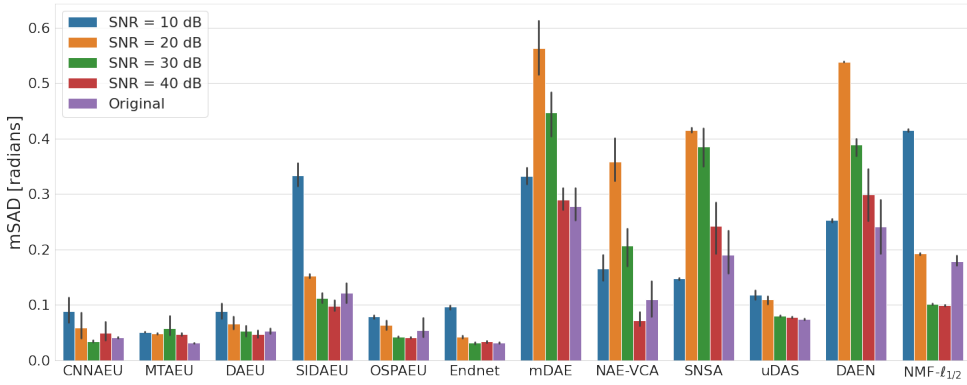


Figure 5.13: A bar plot of the mSAD score for all methods for the Samson dataset with four different levels of SNR and the original uncorrupted dataset. The black vertical lines show the standard deviation. All experiments consisted of 25 runs.

5.1.6 Computation cost

Table 5.8: The computation time in seconds for a single run for the methods for three datasets.

Dataset	Urban (4 endm.)	Houston	Apex
CNNAEU	95	24	347
MTAEU	58	55	60
DAEU	79	77	83
SIDAEU	47	44	52
OSPAEU	34	18	34
Endnet	112	98	105
mDAE	27	36	69
NLAEU	220	22	215
DAEN	3400	1102	3602
SNSA	809	708	1813
uDAS	940	158	1654

It is very difficult to meaningfully compare the running time of different methods since it is very dependent upon hyperparameters such as batch size, the number of training data samples, and implementation details such as what programming language and deep learning framework is being used. Also, CNN methods run on GPU while dense neural network methods run on CPU if implemented using Tensorflow or Matlab. Table 5.8 shows running times in seconds for a single run for the methods for three different datasets. The main conclusion that can be drawn from Table 5.8 is that methods implemented in the Python programming language using the Tensorflow or Pytorch deep learning frameworks are significantly faster than the methods

implemented in Matlab. The experiments were performed on a computer having an eight-core CPU and 64 GB of memory and a GPU having 11 GB of memory.

Chapter 6

Conclusions

In this chapter, the conclusions and main contributions are reviewed. Future research topics are also discussed.

6.1 Main Contributions

The goal of this thesis was to develop blind unmixing methods based on deep learning autoencoders and investigate the strength and weakness of these methods. A secondary objective was to find ways to take the spatial correlations between pixels into account to improve the performance of the autoencoder based methods. The main contributions of this thesis are listed here.

6.2 Spatial-Spectral Hyperspectral Unmixing Using Multitask Learning

An autoencoder based method, MTAEU, which directly makes use of the spatial correlation between pixels in an HSI is introduced. The method leverages multitask learning by simultaneously unmixing a neighborhood of pixels at once. The unmixing of each pixel is a separate encoder task but the tasks share a hidden layer and the linear decoder is shared between all the tasks.

In HSIs, the pixels are often highly correlated in terms of their spatial location. While many techniques for HSU do not take advantage of this spatial structure, there are some methods known as spectral-spatial techniques that do. These methods use assumptions about the spatial correlation of pixels in an HSI as a way to constrain the sparsity and smoothness of the resulting abundance maps.

Even though the tasks are identical or doing the same thing, MTL is still beneficial. The main benefits of MTL are the following:

- Faster learning
- Reduced risk of overfitting
- Improved stability
- Incorporation of spatial information

The method was evaluated using two real HSIs and compared to some state-of-the-art methods and was found to perform significantly better than the comparison methods. The results for the two datasets used confirm that MTL

(a) makes the method spatial in nature.

(b) speeds up the convergence of the method.

(c) leads to better consistency.

(d) lowers the variance of estimated endmembers and abundance maps.

The experiments also show that the method does indeed exploit the spatial correlations in an HSI.

6.3 Convolutional Autoencoder For Spectral-Spatial Hyperspectral Unmixing

The first fully 2D convolutional neural network autoencoder method (CNNAEU) for blind hyperspectral unmixing is introduced. Both the encoder and the decoder are 2D CNN models and the method is trained on patches from an HSI to be unmixed. Because the method is trained on patches, the spatial structure is preserved throughout, and abundance maps arise naturally as feature maps produced by the encoder. This makes it particularly easy to implement various spatial regularizations on the abundance maps.

In order to interpret the weights of a decoder having spatial filters larger than 1×1, the following spatial-spectral unmixing model is introduced:G given P observed spectra, each having B bands. Having estimated the number of endmembers in the HSI as R, a novel spectral-spatial model is assumed, given by

$$\boldsymbol{x}_p = \boldsymbol{M}\boldsymbol{s}_p + \sum_{i\in\mathcal{N}_p\setminus p} \boldsymbol{M}_i\tilde{\boldsymbol{s}}_i + \boldsymbol{\epsilon}_p, \quad p = 1, \cdots, P, \tag{6.1}$$

where $\boldsymbol{s}_p \in \mathbb{R}_+^{R\times1}$ is an abundance vector that sums to one, $\boldsymbol{M} \in \mathbb{R}_+^{B\times R}$ contains the endmembers in its columns, $\mathcal{N}_p \setminus p$ is an $f \times f$ neighborhood around p, with f an odd number, while excluding p (the center pixel), $\boldsymbol{M}_i \in \mathbb{R}_+^{B\times R}$, $\tilde{\boldsymbol{s}}_i \in \mathbb{R}_+^{R\times1}$, $\boldsymbol{M}_i\tilde{\boldsymbol{s}}_i$ is the spectral contribution to $\boldsymbol{x}_p$ from location i in its neighborhood, and $\boldsymbol{\epsilon}_p$ is noise. We index the pixels in the neighborhood $\mathcal{N}$ as shown in

This model shows how neighboring pixels contribute to the reconstruction of each pixel's spectrum, and how the method takes advantage of spatial correlations in both the encoder and the decoder.

The CNNAEU method was compared to four conventional and three deep learning state-of-the-art unmixing methods using four real HSIs. Three of the comparison methods are spectral-spatial methods. Experimental results show that the CNNAEU technique outperforms the comparison methods when it comes to endmember extraction. It achieved the best SAD score in all datasets except for Samson dataset, which might be explained by its small size. Overall, the method had very low variance and the best consistency by far. Experiments also demonstrate the method's robustness to noise.

The abundance maps produced by CNNAEU are binary and therefore do not achieve very low MSE from reference. To circumvent this problem, the method was extended to determine the abundance maps directly from the endmembers using a fully connected autoencoder with a linear and fixed decoder. The maps produced by this new method, CNNAEU2, are not as binary looking and achieved the best MSE score for the Apex and the Houston datasets and the second-best score for the other two datasets. Overall, the proposed method significantly outperforms all the other methods and compares very favorably to our previous methods.

6.4 Blind hyperspectral unmixing using autoencoders: A critical comparison

This work is a review of the literature on blind hyperspectral unmixing using autoencoders. The paper gives an introduction to autoencoders and how they can be used for blind unmixing. It discusses all the various types of autoencoders used in the literature, both for blind unmixing and non-blind, i.e., abundance estimation methods. It gives a comprehensive overview of the various methods based on autoencoders for blind unmixing in the literature and has a critical comparison of a variety of methods.

Chapters 1, 2, and 5 of this thesis are based on this paper. Chapter 2 discusses the various types of autoencoders used in the literature and how they can be used for blind unmixing. It also discussed various way to implement the necessary constraints required for unmixing, i.e., ASC and non-negativity. Also, how spatial information can be incorporated into the autoencoder is discussed. The chapter also tries to answer the question of why autoencoders are so versatile and powerful for blind unmixing compared to traditional methods.

In Chapter 5 the performance of 11 blind unmixing methods was evaluated and compared using four real and four synthetic datasets having increasing spectral variability, with reference endmembers and abundance maps as ground truth. The methods were based on neural network autoencoders, and ablation experiments were performed to study the effects of different ways of enforcing the abundance sum-to-one constraint (ASC), different activation functions for the encoder, and linear vs non-linear decoders. Also, the effect of increasing spectral variability on the performance of the methods was investigated.

The performance of the endmember extraction was evaluated using the mean spectral angle distance (mSAD), and the performance of the abundance map generation was evaluated using the root mean squared error (RMSE) measure. The computation costs of the methods were also compared. The results of these experiments showed that the neural network autoencoder methods were able to achieve high accuracy in endmember extraction and abundance map generation compared to traditional methods.

Eleven different autoencoder methods were compared by performing blind unmixing on four different real hyperspectral datasets in the hope of elucidating what works and what does not. It is generally very hard to interpret experiment results in terms of concrete implementation details of neural network methods when they differ in many ways. Still, some important qualities such as the nature of the objective function, leave a discernible mark on the performance of the methods, and these will be discussed below. Because of inherent spectral variability in real HSIs and the inability of the

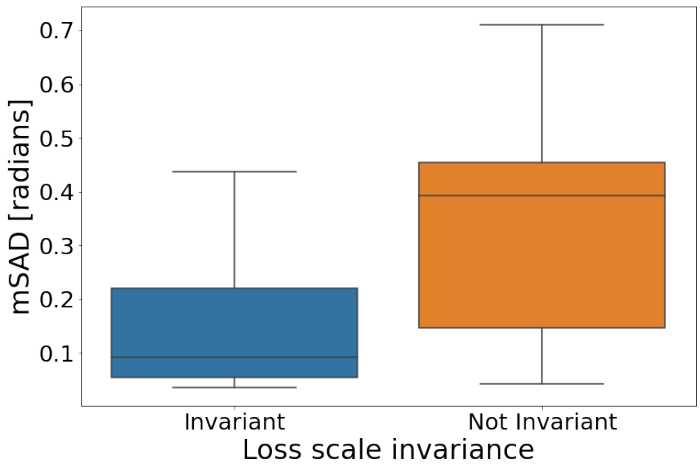


Figure 6.1: Box plot of the average mSAD of methods grouped by whether their loss function is scale invariant, semi-invariant, or not at all.

LMM to model it, the scale invariance of the loss function used by the methods seems to matter greatly. Fig. 6.1 shows the average mSAD of method groups where they are grouped by whether their loss is scale invariant, semi-invariant, or not at all invariant. The SAD and SID similarity measure are scale invariant. The semi-invariant group consists only of the Endnet method, which has a small MSE term in its loss function. All methods using the MSE and the hyper-Laplacian loss function belong to the last group of scale sensitive methods. The graph shows very clearly that having a scale sensitive loss function leads to substantially lower performance than if a scale invariant loss had been used.

This is an important conclusion. Scale sensitive fidelity terms should be avoided if the method is performing unmixing in accordance with the LMM model. Unless the architecture of the method is designed to handle spectral variability, such as the method in [71], a scale invariant loss or as close to it as possible should be chosen. Even if spectral variability is handled by the method, another problem remains if a strongly scale sensitive similarity measure such as MSE is used, which is the inability to extract endmembers which have a very small scale compared to all the other endmembers in a scene, such as the water endmember in the Apex dataset.

Another observation concerns the use of spatial information, i.e., whether methods process a single spectrum at a time, or if they operate on a patch at a time and make use of the spatial correlations existing within real HSIs. The CNNAEU and MTAEU methods perform very well over all, and they are the only methods that operate on whole patches of HSIs at a time. Making use of as much of the available information within an HSI as possible can only lead to better unmixing performance, if done correctly. Better performance can come, e.g., in the form of lowered variance and better consistency. In the light of this, it is somewhat surprising that at the time of writing there is still only one fully convolutional blind unmixing method.

The intersection of hyperspectral unmixing and deep learning is currently a very vibrant field of research, and with more and more autoencoder based method being published with every passing year, utilizing the latest results from the study of autoencoders in the context of DL, one cannot but feel excited for what the future holds.

6.5 Further Work

There are many ways in which the work presented in this thesis can be extended. The following are some possible directions for future work.

- Improve the CNNAEU method to give good abundance maps. The CNNAEU method produces rather binary abundance maps, which is a limitation of the method when using large filters in the decoder and allowing nearby pixels to contribute to the reconstruction of the spectrum of a pixel.
- Develop autoencoder methods that utilize the transformer architecture [130] for blind unmixing. The transformer architecture has been shown to be extremely versatile and dominates the field of natural language processing. The attention mechanism has recently been adapted to images [131] and shows great promise. It would be interesting to see if it can be used for blind unmixing to exploit the local spatial context of each pixel in the HSI to improve the performance.
- One characteristic of current deep learning based unmixing methods is that they are only trained on a single image, the image to be unmixed. This is a limitation of the current methods, and it would be interesting to see if it is possible to train a method on many images and then use it to unmix new images with very

little training or fine-tuning. This could be useful when training models based on the transformer architecture and the training data could be a large collection of HSIs taken with the same sensor.

- Exploring kernel based methods and using encoders based on kernel methods to investigate if the unmixing performance is enhanced by unmixing in the kernelized space.

6.6 Acknowledgements

The author would like to thank Jun Li for the code in [129] and in [83], Jie Chen for the code in [69], and Ying Qu for making the code in [62] available on GitHub.

Bibliography

[1] Alexander F.H. Goetz, Gregg Vane, Jerry E. Solomon, and Barrett N. Rock, "Imaging spectrometry for earth remote sensing," *Science*, vol. 228, no. 4704, pp. 1147–1153, 1985. [Cited on page 2]

[2] Amanda Teixeira Badaró, José Manuel Amigo, Jose Blasco, Nuria Aleixos, Amanda Rios Ferreira, Maria Teresa Pedrosa Silva Clerici, and Douglas Fernandes Barbin, "Near infrared hyperspectral imaging and spectral unmixing methods for evaluation of fiber distribution in enriched pasta," *Food Chemistry*, vol. 343, pp. 128517, 2021. [Cited on page 2]

[3] Tristan D McRae, David Oleksyn, Jim Miller, and Yu-Rong Gao, "Robust blind spectral unmixing for fluorescence microscopy using unsupervised learning," *Plos one*, vol. 14, no. 12, pp. e0225410, 2019. [Cited on page 2]

[4] Chenghai Yang, James H Everitt, and Joe M Bradford, "Airborne hyperspectral imagery and linear spectral unmixing for mapping variation in crop yield," *Precision Agriculture*, vol. 8, no. 6, pp. 279–296, 2007. [Cited on page 2]

[5] Marta B Lopes, Jose M Bioucas-Dias, Mario AT Figueiredo, Jean-Claude Wolff, Nisha Mistry, and John Warrack, "Comparison of near infrared and raman hyperspectral unmixing performances for chemical identification of pharmaceutical tablets," in *2011 3rd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*. IEEE, 2011, pp. 1–4. [Cited on page 2]

[6] Gregory P Asner and Kathleen B Heidebrecht, "Spectral unmixing of vegetation, soil and dry carbon cover in arid regions: comparing multispectral and hyperspectral observations," *International Journal of Remote Sensing*, vol. 23, no. 19, pp. 3939–3958, 2002. [Cited on page 2]

[7] GJ Edelman, E Gaston, TG Van Leeuwen, PJ Cullen, and MCG Aalders, "Hyperspectral imaging for non-contact analysis of forensic traces," *Forensic science international*, vol. 223, no. 1-3, pp. 28–39, 2012. [Cited on page 2]

[8] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006. [Cited on page 2]

[9] Dor Bank, Noam Koenigstein, and Raja Giryes, "Autoencoders," *arXiv e-prints*, vol. cs.LG/2003.05991, 2020. [Cited on page 2]

[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, MIT Press, 2016, `http://www.deeplearningbook.org`. [Cited on page 2]

[11] Giorgio Antonino Licciardi, Xavier Ceamanos, Sylvain Douté, and Jocelyn Chanussot, "Unsupervised nonlinear spectral unmixing by means of nlpca applied to hyperspectral imagery," in *2012 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2012, pp. 1369–1372. [Cited on page 2]

[12] R. Guo, W. Wang, and H. Qi, "Hyperspectral image unmixing using autoencoder cascade," in *2015 7th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, 2015, pp. 1–4. [Cited on pages 2, 10, 11, 12, 40, and 69]

[13] Shen-En Qian, "Hyperspectral satellites, evolution, and development history," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 14, pp. 7032–7056, 2021. [Cited on pages 2 and 3]

[14] Bo-Cai Gao, Marcos J Montes, Curtiss O Davis, and Alexander FH Goetz, "Atmospheric correction algorithms for hyperspectral remote sensing data of land and ocean," *Remote sensing of environment*, vol. 113, pp. S17–S24, 2009. [Cited on page 3]

[15] B.-C. Gao, C. Davis, and A. Goetz, "A review of atmospheric correction techniques for hyperspectral remote sensing of land surfaces and ocean color," in *2006 IEEE International Symposium on Geoscience and Remote Sensing*, 2006, pp. 1979–1981. [Cited on page 4]

[16] R. Heylen, M. Parente, and P. Gader, "A review of nonlinear hyperspectral unmixing methods," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 6, pp. 1844–1868, June 2014. [Cited on page 5]

[17] Pierre-Antoine Thouvenin, Nicolas Dobigeon, and Jean-Yves Tourneret, "Hyperspectral unmixing with spectral variability using a perturbed linear mixing model," *IEEE Transactions on Signal Processing*, vol. 64, no. 2, pp. 525–538, 2016. [Cited on page 5]

[18] L. Drumetz, M. Veganzones, S. Henrot, R. Phlypo, J. Chanussot, and C. Jutten, "Blind hyperspectral unmixing using an extended linear mixing model to address spectral variability," *IEEE Transactions on Image Processing*, vol. 25, no. 8, pp. 3890–3905, 2016. [Cited on page 5]

[19] D. Hong, N. Yokoya, J. Chanussot, and X. X. Zhu, "An augmented linear mixing model to address spectral variability for hyperspectral unmixing," *IEEE Transactions on Image Processing*, vol. 28, no. 4, pp. 1923–1938, 2019. [Cited on page 5]

[20] Ricardo Augusto Borsoi, Tales Imbiriba, and José Carlos Moreira Bermudez, "A data dependent multiscale model for hyperspectral unmixing with spectral variability," *IEEE Transactions on Image Processing*, vol. 29, pp. 3638–3651, 2020. [Cited on page 5]

[21] N. Dobigeon, J. Tourneret, C. Richard, J. C. M. Bermudez, S. McLaughlin, and A. O. Hero, "Nonlinear unmixing of hyperspectral images: Models and algorithms," *IEEE Signal Processing Magazine*, vol. 31, no. 1, pp. 82–94, 2014. [Cited on page 5]

[22] R. Heylen, M. Parente, and P. Gader, "A review of nonlinear hyperspectral unmixing methods," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 6, pp. 1844–1868, 2014. [Cited on page 5]

[23] Y. Altmann, N. Dobigeon, and J. Tourneret, "Bilinear models for nonlinear unmixing of hyperspectral images," in *2011 3rd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, 2011, pp. 1–4. [Cited on page 5]

[24] A. Halimi, Y. Altmann, N. Dobigeon, and J. Tourneret, "Nonlinear unmixing of hyperspectral images using a generalized bilinear model," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 11, pp. 4153–4162, 2011. [Cited on page 5]

[25] J. Sigurdsson, M. O. Ulfarsson, and J. R. Sveinsson, "Blind sparse nonlinear hyperspectral unmixing using an ℓ_q penalty," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 12, pp. 1907–1911, 2018. [Cited on page 5]

[26] Bruce Hapke, "Bidirectional reflectance spectroscopy: 1. theory," *Journal of Geophysical Research: Solid Earth*, vol. 86, no. B4, pp. 3039–3054, 1981. [Cited on page 5]

[27] Chein-I Chang and Qian Du, "Estimation of number of spectrally distinct signal sources in hyperspectral imagery," *IEEE Transactions on geoscience and remote sensing*, vol. 42, no. 3, pp. 608–619, 2004. [Cited on page 6]

[28] J.M. Bioucas-Dias and J.M.P. Nascimento, "Hyperspectral subspace identification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 46, no. 8, pp. 2435–2445, Aug. 2008. [Cited on pages 6 and 41]

[29] Bin Luo, Jocelyn Chanussot, Sylvain Douté, and Liangpei Zhang, "Empirical automatic estimation of the number of endmembers in hyperspectral images," *IEEE Geoscience and Remote Sensing Letters*, vol. 10, no. 1, pp. 24–28, 2012. [Cited on page 6]

[30] B. Rasti, M. O. Ulfarsson, and J. R. Sveinsson, "Hyperspectral subspace identification using SURE," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 12, pp. 2481–2485, Dec 2015. [Cited on pages 6 and 41]

[31] J. M. Bioucas-Dias, A. Plaza, N. Dobigeon, M. Parente, Q. Du, P. Gader, and J. Chanussot, "Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 5, no. 2, pp. 354–379, April 2012. [Cited on pages 6 and 7]

[32] M. D. Iordache, J. M. Bioucas-Dias, and A. Plaza, "Sparse unmixing of hyperspectral data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 6, pp. 2014–2039, June 2011. [Cited on page 6]

[33] M. D. Iordache, J. M. Bioucas-Dias, and A. Plaza, "Total variation spatial regularization for sparse hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 11, pp. 4484–4502, Nov 2012. [Cited on page 6]

[34] M. D. Iordache, J. M. Bioucas-Dias, and A. Plaza, "Collaborative sparse regression for hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 1, pp. 341–354, Jan 2014. [Cited on page 6]

[35] M. D. Iordache, J. M. Bioucas-Dias, A. Plaza, and B. Somers, "MUSIC-CSR: Hyperspectral unmixing via multiple signal classification and collaborative sparse regression," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 7, pp. 4364–4382, July 2014. [Cited on page 6]

[36] J. M. Nascimento and J. M. Dias, "Vertex component analysis: A fast algorithm to unmix hyperspectral data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 4, pp. 898–910, 2005. [Cited on pages 7 and 28]

[37] Michael E Winter, "N-findr: an algorithm for fast autonomous spectral endmember determination in hyperspectral data," in *SPIE's International Symposium on Optical Science, Engineering, and Instrumentation*. International Society for Optics and Photonics, 1999, pp. 266–275. [Cited on page 7]

[38] 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 Transactions on Geoscience and Remote Sensing*, vol. 53, no. 9, pp. 5067–5082, Sept 2015. [Cited on page 7]

[39] Y. Qian, S. Jia, J. Zhou, and A. Robles-Kelly, "Hyperspectral unmixing via $l_{1/2}$ sparsity-constrained nonnegative matrix factorization," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 11, pp. 4282–4297, Nov 2011. [Cited on pages 7, 28, 45, and 69]

[40] J. Sigurdsson, M. O. Ulfarsson, and J. R. Sveinsson, "Hyperspectral unmixing with l_q regularization," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 11, pp. 6793–6806, Nov 2014. [Cited on page 7]

[41] J. M. P. Nascimento and J. M. Bioucas-Dias, "Hyperspectral unmixing based on mixtures of Dirichlet components," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 3, pp. 863–878, March 2012. [Cited on page 7]

[42] N. Dobigeon, S. Moussaoui, M. Coulon, J. Y. Tourneret, and A. O. Hero, "Joint Bayesian endmember extraction and linear unmixing for hyperspectral imagery," *IEEE Transactions on Signal Processing*, vol. 57, no. 11, pp. 4355–4368, Nov 2009. [Cited on page 7]

[43] 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 Transactions on Image Processing*, vol. 19, no. 6, pp. 1403–1413, June 2010. [Cited on page 7]

[44] J. Sigurdsson, M. O. Ulfarsson, and J. R. Sveinsson, "Blind hyperspectral unmixing using total variation and ℓ_q sparse regularization," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 11, pp. 6371–6384, Nov 2016. [Cited on pages 7, 25, and 40]

[45] L. Miao and H. Qi, "Endmember extraction from highly mixed data using minimum volume constrained nonnegative matrix factorization," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 45, no. 3, pp. 765–777, March 2007. [Cited on page 7]

[46] L. Zhang, W. Wei, Y. Zhang, F. Li, and H. Yan, "Structured sparse BAYESIAN hyperspectral compressive sensing using spectral unmixing," in *6th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, June 2014, pp. 1–4. [Cited on page 7]

[47] R. T. Albayrak, A. C. Gurbuz, and B. Gunyel, "Compressed sensing based hyperspectral unmixing," in *22nd Signal Processing and Communications Applications Conference (SIU)*, April 2014, pp. 1438–1441. [Cited on page 7]

[48] Xusu, "Compressive sensing for endmember extraction," in *2nd IEEE International Conference on Computer and Communications (ICCC)*, Oct 2016, pp. 1345–1348. [Cited on page 7]

[49] Feiyun Zhu, "Spectral unmixing datasets with ground truths," *CoRR*, vol. abs/1708.05125, 2017. [Cited on page 8]

[50] Lee J. Rickard, Robert W. Basedow, Edward F. Zalewski, Peter R. Silverglate, and Mark Landers, "HYDICE: an airborne system for hyperspectral imaging," in *Imaging Spectrometry of the Terrestrial Environment*, Gregg Vane, Ed. International Society for Optics and Photonics, 1993, vol. 1937, pp. 173 – 179, SPIE. [Cited on page 8]

[51] Michael E. Schaepman, Michael Jehle, Andreas Hueni, Petra D'Odorico, Alexander Damm, Jürg Weyermann, Fabian D. Schneider, Valérie Laurent, Christoph Popp, Felix C. Seidel, Karim Lenhard, Peter Gege, Christoph Kuchler, Jason Brazile, Peter Kohler, Lieve De Vos, Koen Meuleman, Roland Meynart, Daniel Schläpfer, Mathias Kneubühler, and Klaus I. Itten, "Advanced radiometry measurements and earth science applications with the airborne prism experiment (apex)," *Remote Sensing of Environment*, vol. 158, pp. 207 – 219, 2015. [Cited on page 9]

[52] Chen Shi and Le Wang, "Incorporating spatial information in spectral unmixing: A review," *Remote Sensing of Environment*, vol. 149, pp. 70 – 87, 2014. [Cited on pages 10 and 40]

[53] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical Evaluation of Rectified Activations in Convolutional Network," *ArXiv e-prints*, May 2015. [Cited on pages 3 and 27]

[54] Matan Benyamin, Hadar Genish, Ran Califa, Lauren Wolbromsky, Michal Ganani, Zhen Wang, Shuyun Zhou, Zheng Xie, and Zeev Zalevsky, "Autoencoder based blind source separation for photoacoustic resolution enhancement," *Scientific Reports*, vol. 10, pp. 1–7, 12 2020. [Cited on page 4]

[55] K. H. Tsai, W. C. Wang, C. H. Cheng, C. Y. Tsai, J. K. Wang, T. H. Lin, S. H. Fang, L. C. Chen, and Y. Tsao, "Blind monaural source separation on heart and lung sounds based on periodic-coded deep autoencoder," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 11, pp. 3203–3214, 2020. [Cited on page 4]

[56] M. Mikulski and J. Duda, "Toroidal autoencoder," *ArXiv*, vol. abs/1903.12286, 2019. [Cited on page 4]

[57] Diederik P Kingma and Max Welling, "Auto-encoding variational bayes," *stat*, vol. 1050, pp. 10, 2014. [Cited on pages 4 and 5]

[58] Solomon Kullback and Richard A Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951. [Cited on page 5]

[59] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and I. Goodfellow, "Adversarial autoencoders," *ArXiv*, vol. abs/1511.05644, 2015. [Cited on pages 5 and 6]

[60] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion.," *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, 2010. [Cited on page 6]

[61] Minmin Chen, E. Xu, and Kilian Q. Weinberger, "Marginalized stacked denoising autoencoders," 2012. [Cited on page 7]

[62] Ying Qu and Hairong Qi, "uDAS: An untied denoising autoencoder with sparsity for spectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 3, pp. 1698–1712, 2019. [Cited on pages 7, 10, 11, 12, 29, 40, 45, 69, and 98]

[63] Jonathan Masci, Ueli Meier, Dan Cireşan, and Jürgen Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," in *International conference on artificial neural networks*. Springer, 2011, pp. 52–59. [Cited on page 7]

[64] B. Palsson, M. O. Ulfarsson, and J. R. Sveinsson, "Convolutional autoencoder for spectral-spatial hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, pp. 1–15, 2020. [Cited on pages 7, 8, 10, 11, 12, 16, and 69]

[65] H. Mhaskar, Q. Liao, and T. Poggio, "Learning functions: When is deep better than shallow," *arXiv: Learning*, 2016. [Cited on page 8]

[66] B. Palsson, J. Sigurdsson, J. R. Sveinsson, and M. O. Ulfarsson, "Hyperspectral unmixing using a neural network autoencoder," *IEEE Access*, vol. 6, pp. 25646–25656, 2018. [Cited on pages 8, 10, 11, 12, 24, 29, 40, 45, and 69]

[67] B. Palsson, J. R. Sveinsson, and M. O. Ulfarsson, "Spectral-spatial hyperspectral unmixing using multitask learning," *IEEE Access*, vol. 7, pp. 148861–148872, 2019. [Cited on pages 8, 10, 11, 12, 14, 16, 40, 45, and 69]

[68] Y. Su, X. Xu, J. Li, H. Qi, P. Gamba, and A. Plaza, "Deep autoencoders with multitask learning for bilinear hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, pp. 1–15, 2020. [Cited on pages 8, 11, 12, and 14]

[69] M. Wang, M. Zhao, J. Chen, and S. Rahardja, "Nonlinear unmixing of hyperspectral data via deep autoencoder networks," *IEEE Geoscience and Remote Sensing Letters*, vol. 16, no. 9, pp. 1467–1471, 2019. [Cited on pages 8, 10, 11, 12, 40, 69, 73, and 98]

[70] Z. Dou, K. Gao, X. Zhang, H. Wang, and J. Wang, "Hyperspectral unmixing using orthogonal sparse prior-based autoencoder with hyper-laplacian loss and data-driven outlier detection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 9, pp. 6550–6564, 2020. [Cited on pages 8, 10, 11, 12, 13, and 69]

[71] Z. Dou, K. Gao, X. Zhang, H. Wang, and J. Wang, "Blind hyperspectral unmixing using dual branch deep autoencoder with orthogonal sparse prior," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 2428–2432. [Cited on pages 8, 10, 12, and 97]

[72] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015. [Cited on pages 9, 27, and 42]

[73] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Mądry, "How does batch normalization help optimization?," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2018, NIPS'18, p. 2488–2498, Curran Associates Inc. [Cited on page 9]

[74] Nitish Shirish Keskar, Jorge Nocedal, Ping Tak Peter Tang, Dheevatsa Mudigere, and Mikhail Smelyanskiy, "On large-batch training for deep learning: Generalization gap and sharp minima," 2017, 5th International Conference on Learning Representations, ICLR 2017 ; Conference date: 24-04-2017 Through 26-04-2017. [Cited on page 9]

[75] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng, "Rectifier nonlinearities improve neural network acoustic models," . [Cited on page 9]

[76] Y. Su, A. Mariononi, J. Li, A. Plaza, and P. Gamba, "Nonnegative sparse autoencoder for robust endmember extraction from remotely sensed hyperspectral images," in *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, July 2017. [Cited on pages 10 and 11]

[77] S. Ozkan, B. Kaya, and G. B. Akar, "Endnet: Sparse autoencoder network for endmember extraction and hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 1, pp. 482–496, Jan 2019. [Cited on pages 10, 11, 12, 13, 40, and 69]

[78] Y. Qu, R. Guo, and H. Qi, "Spectral unmixing through part-based non-negative constraint denoising autoencoder," in *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, July 2017. [Cited on pages 10, 11, 12, and 13]

[79] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014. [Cited on pages 10 and 27]

[80] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler, "Efficient object localization using convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 648–656. [Cited on pages 10 and 43]

[81] F. Palsson, J. Sigurdsson, J. R. Sveinsson, and M. O. Ulfarsson, "Neural network hyperspectral unmixing with spectral information divergence objective," in *2017 IEEE International Geoscience and Remote Sensing Symposium - IGARSS*, July 2017. [Cited on pages 11 and 69]

[82] Y. Su, A. Marinoni, J. Li, J. Plaza, and P. Gamba, "Stacked nonnegative sparse autoencoders for robust hyperspectral unmixing," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 9, pp. 1427–1431, Sep. 2018. [Cited on pages 11, 12, 14, and 29]

[83] Y. Su, J. Li, A. Plaza, A. Marinoni, P. Gamba, and S. Chakravortty, "Daen: Deep autoencoder networks for hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 7, pp. 4309–4321, July 2019. [Cited on pages 11, 13, 14, 15, 40, 69, and 98]

[84] B. Yan, Z. Wu, H. Liu, Y. Xu, and Z. Wei, "Hyperspectral unmixing via wavelet based autoencoder network," in *2019 10th Workshop on Hyperspectral Imaging and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, 2019, pp. 1–5. [Cited on pages 11, 12, and 13]

[85] Chris Ding, Ding Zhou, Xiaofeng He, and Hongyuan Zha, "R1-pca: Rotational invariant l1-norm principal component analysis for robust subspace factorization," in *Proceedings of the 23rd International Conference on Machine Learning*, New York, NY, USA, 2006, ICML '06, p. 281–288, Association for Computing Machinery. [Cited on page 13]

[86] Rich Caruana, "Multitask learning," *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997. [Cited on pages 14, 16, 23, 24, and 25]

[87] Sebastian Ruder, "An overview of multi-task learning in deep neural networks," *arXiv preprint arXiv:1706.05098*, 2017. [Cited on pages 14, 16, and 23]

[88] Jonathan Baxter, "A bayesian/information theoretic model of learning to learn via multiple task sampling," *Machine learning*, vol. 28, no. 1, pp. 7–39, 1997. [Cited on pages 14 and 24]

[89] Ricardo Augusto Borsoi, Tales Imbiriba, and José Carlos Moreira Bermudez, "Deep generative endmember modeling: An application to unsupervised spectral unmixing," *IEEE Transactions on Computational Imaging*, vol. 6, pp. 374–384, 2020. [Cited on page 15]

[90] Anyou Min, Ziyang Guo, Hong Li, and Jiangtao Peng, "Jmnet: Joint metric neural network for hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, pp. 1–12, 2021. [Cited on page 15]

[91] Savas Ozkan and Gozde Bozdagi Akar, "Spectral unmixing with multinomial mixture kernel and wasserstein generative adversarial loss," *arXiv e-prints*, pp. arXiv–2012, 2020. [Cited on page 15]

[92] Min Zhao, Mou Wang, Jie Chen, and Susanto Rahardja, "Perceptual loss-constrained adversarial autoencoder networks for hyperspectral unmixing," *IEEE Geoscience and Remote Sensing Letters*, vol. 19, pp. 1–5, 2022. [Cited on page 15]

[93] Min Zhao, Mou Wang, Jie Chen, and Susanto Rahardja, "Hyperspectral unmixing for additive nonlinear models with a 3-d-cnn autoencoder network," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–15, 2022. [Cited on pages 16 and 73]

[94] Lin Qi, Feng Gao, Junyu Dong, Xinbo Gao, and Qian Du, "Sscu-net: Spatial–spectral collaborative unmixing network for hyperspectral images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–15, 2022. [Cited on page 16]

[95] Baohua Jin, Sijia Li, Wei Huang, and Yunfei Zhu, "Spatial-spectral dual-branch autoencoder based on adaptive convolution for hyperspectral unmixing," in *2023 4th International Conference on Computer Engineering and Application (ICCEA)*, 2023, pp. 940–944. [Cited on page 16]

[96] Kazi Tanzeem Shahid and Ioannis D. Schizas, "Spatial-aware hyperspectral nonlinear unmixing autoencoder with endmember number estimation," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 15, pp. 20–41, 2022. [Cited on pages 17 and 73]

[97] Thomas Hofmann, Bernhard Schölkopf, and Alexander J. Smola, "Kernel methods in machine learning," *The Annals of Statistics*, vol. 36, no. 3, pp. 1171 – 1220, 2008. [Cited on page 17]

[98] Matthias Feurer and Frank Hutter, "Hyperparameter optimization," in *Automated machine learning*, pp. 3–33. Springer, Cham, 2019. [Cited on page 18]

[99] James Bergstra and Yoshua Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. 10, pp. 281–305, 2012. [Cited on page 18]

[100] Jasper Snoek, Hugo Larochelle, and Ryan P Adams, "Practical bayesian optimization of machine learning algorithms," *Advances in neural information processing systems*, vol. 25, 2012. [Cited on page 18]

[101] Marcelo Pereyra, José M Bioucas-Dias, and Mário AT Figueiredo, "Maximum-a-posteriori estimation with unknown regularisation parameters," in *2015 23rd European Signal Processing Conference (EUSIPCO)*. IEEE, 2015, pp. 230–234. [Cited on page 18]

[102] Savas Ozkan and Gozde Bozdagi Akar, "Deep spectral convolution network for hyperspectral unmixing," in *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 3313–3317. [Cited on page 20]

[103] Behnood Rasti, Bikram Koirala, Paul Scheunders, and Pedram Ghamisi, "Undip: Hyperspectral unmixing using deep image prior," *IEEE Transactions on Geoscience and Remote Sensing*, pp. 1–15, 2021. [Cited on page 20]

[104] Lin Qi, Jie Li, Ying Wang, Mingyu Lei, and Xinbo Gao, "Deep spectral convolution network for hyperspectral image unmixing with spectral library," *Signal Processing*, vol. 176, pp. 107672, 2020. [Cited on page 20]

[105] Zhu Han, Danfeng Hong, Lianru Gao, Bing Zhang, and Jocelyn Chanussot, "Deep half-siamese networks for hyperspectral unmixing," *IEEE Geoscience and Remote Sensing Letters*, pp. 1–5, 2020. [Cited on page 20]

[106] Danfeng Hong, Lianru Gao, Jing Yao, Naoto Yokoya, Jocelyn Chanussot, Uta Heiden, and Bing Zhang, "Endmember-guided unmixing network (egu-net): A general deep learning framework for self-supervised hyperspectral unmixing," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2021. [Cited on page 20]

[107] Qiwen Jin, Yong Ma, Xiaoguang Mei, Hao Li, and Jiayi Ma, "Utdn: An unsupervised two-stream dirichlet-net for hyperspectral unmixing," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 1885–1889. [Cited on page 20]

[108] Jianhui Chen, Jiayu Zhou, and Jieping Ye, "Integrating low-rank and group-sparse structures for robust multi-task learning," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 42–50. [Cited on page 24]

[109] M. Iordache, J. M. Bioucas-Dias, and A. Plaza, "Total variation spatial regularization for sparse hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 11, pp. 4484–4502, Nov 2012. [Cited on pages 25 and 40]

[110] Xiangrong Zhang, Chen Li, Jingyan Zhang, Qimeng Chen, Jie Feng, Licheng Jiao, and Huiyu Zhou, "Hyperspectral unmixing via low-rank representation with space consistency constraint and spectral library pruning," *Remote Sensing*, vol. 10, no. 2, pp. 339, 2018. [Cited on pages 25 and 40]

[111] S. Zhang, J. Li, H. Li, C. Deng, and A. Plaza, "Spectral - spatial weighted sparse regression for hyperspectral image unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 6, pp. 3265–3276, June 2018. [Cited on pages 25 and 40]

[112] J. Sigurdsson, M. O. Ulfarsson, J. R. Sveinsson, and J. A. Benediktsson, "Smooth spectral unmixing using total variation regularization and a first order roughness penalty," in *2013 IEEE International Geoscience and Remote Sensing Symposium - IGARSS*, 2013, pp. 2160–2163. [Cited on pages 25 and 40]

[113] J. Sigurdsson, M. O. Ulfarsson, and J. R. Sveinsson, "Total variation and ℓ_q based hyperspectral unmixing for feature extraction and classification," in *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, July 2015, pp. 437–440. [Cited on pages 25 and 40]

[114] A. Huck and M. Guillaume, "Robust hyperspectral data unmixing with spatial and spectral regularized nmf," in *2010 2nd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing*, June 2010, pp. 1–4. [Cited on pages 25 and 40]

[115] J. Liu, J. Zhang, Y. Gao, C. Zhang, and Z. Li, "Enhancing spectral unmixing by local neighborhood weights," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 5, no. 5, pp. 1545–1552, Oct 2012. [Cited on pages 25 and 40]

[116] X. Zhang, J. Zhang, C. Li, C. Cheng, L. Jiao, and H. Zhou, "Hybrid unmixing based on adaptive region segmentation for hyperspectral imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 7, pp. 3861–3875, July 2018. [Cited on pages 25 and 40]

[117] Xu Sun, Feifei Zhang, Lina Yang, Bing Zhang, and Lianru Gao, "A hyperspectral image spectral unmixing method integrating slic superpixel segmentation," in *2015 7th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*. IEEE, 2015, pp. 1–4. [Cited on pages 25 and 41]

[118] Jiarui Yi and Miguel Velez-Reyes, "Dimensionality reduction using superpixel segmentation for hyperspectral unmixing using the cNMF," in *Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XXIII*. International Society for Optics and Photonics, 2017, vol. 10198, p. 101981H. [Cited on pages 25 and 41]

[119] Xiang Xu, Jun Li, Changshan Wu, and Antonio Plaza, "Regional clustering-based spatial preprocessing for hyperspectral unmixing," *Remote Sensing of Environment*, vol. 204, pp. 333–346, 2018. [Cited on pages 25 and 41]

[120] Xinyu Wang, Yanfei Zhong, Liangpei Zhang, and Yanyan Xu, "Spatial Group Sparsity Regularized Nonnegative Matrix Factorization for Hyperspectral Unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 11, pp. 6287–6304, Nov 2017. [Cited on pages 25, 29, 41, and 45]

[121] R. Mittelman, N. Dobigeon, and A. O. Hero, "Hyperspectral image unmixing using a multiresolution sticky HDP," *IEEE Transactions on Signal Processing*, vol. 60, no. 4, pp. 1656–1671, April 2012. [Cited on pages 29 and 45]

[122] T. Tieleman and G. Hinton, "Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude," COURSERA: Neural Networks for Machine Learning, 2012. [Cited on page 29]

[123] Y. Su, A. Marinoni, J. Li, A. Plaza, and P. Gamba, "Nonnegative sparse autoencoder for robust endmember extraction from remotely sensed hyperspectral images," in *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, July 2017, pp. 205–208. [Cited on page 40]

[124] Le Wang, Chen Shi, Chunyuan Diao, Wenjie Ji, and Dameng Yin, "A survey of methods incorporating spatial information in image classification and spectral unmixing," *International Journal of Remote Sensing*, vol. 37, no. 16, pp. 3870–3910, 2016. [Cited on page 40]

[125] M. C. Torres-Madronero and M. Velez-Reyes, "Integrating spatial information in unsupervised unmixing of hyperspectral imagery using multiscale representation," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 6, pp. 1985–1993, June 2014. [Cited on page 40]

[126] S. Mei, M. He, Y. Zhang, Z. Wang, and D. Feng, "Improving spatial – spectral endmember extraction in the presence of anomalous ground objects," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 11, pp. 4210–4222, Nov 2011. [Cited on page 40]

[127] M. Zortea and A. Plaza, "Spatial preprocessing for endmember extraction," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 47, no. 8, pp. 2679–2693, Aug 2009. [Cited on page 40]

[128] D. C. Heinz and Chein-I-Chang, "Fully constrained least squares linear spectral mixture analysis method for material quantification in hyperspectral imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 39, no. 3, pp. 529–545, March 2001. [Cited on page 44]

[129] Y. Su, A. Marinoni, J. Li, J. Plaza, and P. Gamba, "Stacked nonnegative sparse autoencoders for robust hyperspectral unmixing," *IEEE Geoscience and Remote Sensing Letters*, pp. 1–5, 2018. [Cited on pages 45, 69, and 98]

[130] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017. [Cited on page 97]

[131] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al., "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020. [Cited on page 97]