Article

Tensor-based Factorization Algorithms for Pixel-wise Classification of Hyperspectral Data Using Deep Convolutional Networks

Josué López [1,*], Deni Torres [1] and Clement Atzberger [3]

[1] Center for Research and Advanced Studies of the National Polytechnic Institute, Telecommunications Group, Av del Bosque 1145, Zapopan 45017, Mexico; dtorres@gdl.cinvestav.mx

[3] University of Natural Resources and Life Science, Institute of Geomatics, Peter Jordan 82, Vienna 1180, Austria; clement.atzberger@boku.ac.at

* Correspondence: josue.lopez@cinvestav.mx

Version October 2, 2020 submitted to Remote Sens.

Abstract: Tensor-based algorithms for data compression have evolved in recent years according to the needs of several research areas. Tucker Decomposition (TKD) is one of the most popular factorization methods based on tensor algebra, but it is clear that it is not the only algorithm wich can produce a factorization for a given input data set. Besides, this decomposition does not have singular solutions, i.e., it converges to local minima. Hence, depending on the input data, tensor-based decompositions can achieve better solution to a specific input. The phenomenology of Remote Sensing (RS) Hyperspectral Images (HSI) belongs to the set of natural numbers, i.e., the set of positive integers. Hence, a non-negative tensor factorization suggest a more suitable decomposition for positive data by nature. The main purpose of this work is to prove the benefits in processing time, as well as in accuracy, of using a well-posed factorization algorithm. Specifically, this paper performs a quantitative analysis of tensor-based factorization algorithms applied to semantic segmentation of HSI using Deep Convolutional Networks (DCN).

Keywords: deep convolutional networks; hyperspectral imagery; tensor decomposition

1. Introduction

Big data compression has been one of the most active research areas in recente years. The insertion of tensor-based algorithms for this sort of tasks drove a revolution in several areas such as image processing.

HSI

ML

This work

1.1. State of art

There are several works focused on the development of frameworks that reduce execution time of machine learning algorithms for semantic segmentation of hyperspectral data sets []. The crucial factor, which is addressed in this work, is to achieve compression of the input data to reduce the high number of computatios, but without sacrificing precision in the classification task.

MODIFY!!! In recent years, spectral data for earth surface classification has been a very active research area. Methods proposed by Kemker et al. [11,20], Hamida et al. [21], and López et al. [18] use CNNs for RS-CNNMSI pixel-wise classification. Nevertheless, processing raw spectral data with deep learning (DL) algorithms is computationally very expensive. Wang et al. [22] introduced a salient band selection method for HSIs by manifold ranking, and Li et al. [23] proposed a band selection method from the perspective of spectral shape similarity analysis of RS-HSIs to obtain less computational complexity. However, some surface materials differentiate from each other in specific bands, so cutting off spectral bands negatively affected further classification tasks.

More recently, the use of tensor approach for spectral images compression has been introduced; see Zhang et al. [24]. Many authors adopted dimensionality reduction algorithms, such as PCA [16] and **SVD!** (**SVD!**), for spectral image compression. Other authors have made efforts to reduce the computational cost in CNNs for image classification by using TD algorithms [25,26]. Astrid et al. in [25] proposed a CNN compression method based on CPD and the tensor power method where they achieved significant reduction in memory and computational cost. Chien et al. in [26] presents a tensor-factorized ANN, which integrates TD and ANNs for multi-way feature extraction and classification. Nevertheless, although the idea is to compress data in order to reduce computational cost and processing time, these works compress or decompose the data of the hyper-parameters within the network, which causes the training of the semantic segmentation or classification network to be slower due to the change of the weights in the tensor decomposition.

An et al. [27] proposed an unsupervised tensor-based multiscale low rank decomposition (T-MLRD) method for hyperspectral image dimensionality reduction, and Li et al. in [28] proposed a low-complexity compression approach for multispectral images based on convolution neural networks CNNs with nonnegative Tucker decomposition (NTD). Nevertheless, these methods reduce the tensor in every dimension, which is self-defeating for a segmentation CNN. Besides, the non-negative decomposed tensor proposed in [28] causes slower convergence in DL algorithms. In [29] An et al. proposed a tensor discriminant analysis (TDA) model via compact feature representation, wherein the traditional linear discriminant analysis was extended to tensor space to make the resulting feature representation more discriminant. However, this approach still leads to a degradation of the spatial resolution, which disturbed the CNN performance.

Table 1 summarizes some of the most cited related papers, wich deal with the compression-classification issue.

Table 1. Related work in spectral imagery semantic segmentation.

Reference	Input	Decomposition	Reduction	Classifier
Li, S. et al. [23] (2014)	HSI	-	Band selection	SVM
Zhang, L. et al. [24] (2015)	HSI	TKD	Spatial-Spectral	-
Wan, Q. et al. [22] (2016)	HSI	-	Band selection	SVM/kNN/CART
Kemke, R. et al. [11] (2017)	MSI	-	-	CNN
Hamida, A. et al. [21] (2017)	MSI	-	-	CNN
Li, J. et al. [28] (2019)	MSI	NTD-CNN	Spatial-spectral	-
An, J. et al. [27] (2019)	HSI	T-MLRD	Spatial-spectral	SVM/1NN
An, J. et al. [29] (2019)	HSI	TDA	Spatial-spectral	SVM/1NN
Our framework (2019)	MSI	HOOI	Spectral	FCN

1.2. Contribution

Compared with the existing methods in the state of art, this work proposes a framework that reduces computational load in semantic segmentation CNNs by finding lower dimensionality representations of materials in RS spectral images through TD. This framework preserves spatial dimensions, while reducing spectral data, producing new low correlated spectral or tensor bands, so that a CNN is able to classify each input pixel to a class in much lower computational time and keeping high performance metrics. Different to [28] we propose a transformation of the spectral images by TKD, which does not limit the decomposition to non-negative values. Comparing with [27] our framework preserves spatial dimension to keep high accuracy in the semantic segmentation algorithm.

Tensor-based decomposition produces better data representations of the original data and the elements in the decomposition present high orthogonality degree.

We can summarize the contribution of this work with the following two points:

1.

The remainder of this work is organized as follows. Section **??** introduces tensor algebra notation and basic concepts to familiarize the reader with the symbology used in this paper. Section **??** presents the problem statement of this work and the mathematical definition. In Section 4, CNN theory is described for classification and semantic segmentation. Section **??** presents the framework proposed for compression and semantic segmentation of spectral images. Experimental results are presented in Section **??**. Finally, Sections **??** and 8 present a discussion and conclusions based on the results obtained in the experiments.

2. Tensor-Based Factorizations

In this paper, to express and develop the mathematical model of the work, the notation used by Kolda et al. [17] has been used, which has been accepted in several recent articles as a conventional notation. The following table summarizes the notation in a way that makes the following sections easier to read.

Table 2. Tensor algebra notation summary

$\boldsymbol{\mathcal{A}}$, $\mathbf{A}$, $\mathbf{a}$, a	Tensor, matrix, vector and scalar respectively
$\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$	N-order tensor of size $I_1 \times \cdots \times I_N$.
$a_{i_1 \ldots i_N}$	An element of a tensor
$\mathbf{a}_{:i_2 i_3}$, $\mathbf{a}_{i_1 : i_3}$, and $\mathbf{a}_{i_1 i_2 :}$	Column, row and tube fibers of a third order tensor
$\mathbf{A}_{i_1 ::}$, $\mathbf{A}_{: i_2 :}$, $\mathbf{A}_{:: i_3}$	Horizontal, lateral and frontal slices for a third order tensor
$\mathbf{A}^{(n)}$, $\mathbf{a}^{(n)}$	A matrix/vector element from a sequence of matrices/vectors
$\mathbf{A}_{(n)}$	Mode-n matricization of a tensor. $\mathbf{A}_{(n)} \in \mathbb{R}^{I_n \times \prod_{m \neq n} I_m}$
$\boldsymbol{\mathcal{X}} = \mathbf{a}^{(\mathbf{1})} \circ \cdots \circ \mathbf{a}^{(\mathbf{N})}$	Outer product of N vectors, where $x_{i_1 i_2 \ldots i_N} = a^{(1)}_{i_1} \ldots a^{(N)}_{i_N}$
$\langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \rangle$	Inner product of two tensors.
$\boldsymbol{\mathcal{B}} = \boldsymbol{\mathcal{A}} \times_n \mathbf{U}$	n-mode product of tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ by a matrix $\mathbf{U} \in \mathbb{R}^{J \times I_n}$ along axis n.

It is also necessary to introduce some tensor algebra operations and basic concepts used in later explanations. These notations were taken textually from [17].

2.1. Matricization

The mode-n matricization is the process of reordering the elements of a tensor into a matrix along axis n and it is denoted as $\mathbf{A}_{(n)} \in \mathbb{R}^{I_n \times \prod_{m \neq n} I_m}$.

2.2. Outer Product

The outer product of N vectors $\boldsymbol{\mathcal{X}} = \mathbf{a}^{(\mathbf{1})} \circ \cdots \circ \mathbf{a}^{(\mathbf{N})}$ produces a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ where $\circ$ denotes the outer product and $\mathbf{a}^{(\mathbf{n})}$ denotes a vector in a sequence of N vectors and each element of the tensor is the product of the corresponding vector elements; i.e., $x_{i_1 i_2 \ldots i_N} = a^{(1)}_{i_1} \ldots a^{(N)}_{i_N}$.

2.3. Inner Product

The inner product of two tensors $\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ is the sum of the products of their entries; i.e., $\langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \rangle = \sum_{i_1=1}^{I_1} \cdots \sum_{i_N=1}^{I_N} a_{i_1 \ldots i_N} b_{i_1 \ldots i_N}$.

2.4. N-Mode Product

It means the multiplication of a tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ by a matrix $\mathbf{U} \in \mathbb{R}^{J \times I_n}$ or vector $\mathbf{u} \in \mathbb{R}^{I_n}$ in mode n; i.e., along axis n. It is represented by $\boldsymbol{\mathcal{B}} = \boldsymbol{\mathcal{A}} \times_n \mathbf{U}$, where $\boldsymbol{\mathcal{B}} \in \mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N}$ [17].

2.5. Rank-One Tensor

A tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ is rank one if it can be written as the outer product of N vectors; i.e., $\mathcal{X} = \mathbf{a}^{(\mathbf{1})} \circ \cdots \circ \mathbf{a}^{(\mathbf{N})}$.

2.6. Rank-R Tensor

The rank of a tensor rank$(\mathcal{X})$ is the smallest number of components in a CPD; i.e., the smallest number of rank-one tensors that generate $\mathcal{X}$ as their sum [17].

2.7. N-Rank

The n-rank of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ denoted rank$_n(\mathcal{X})$, is the column rank of $\mathbf{X}_{(n)}$; i.e., the dimension of the vector space spanned by the mode-n fibers. Hence, if $R_n \equiv \text{rank}_n(\mathcal{X})$ for $n = 1, \ldots, N$, we can say that $\mathcal{X}$ has a rank $- (R_1, \ldots, R_N)$ tensor.

All the tensor algebra notation presented until this point is summarized in Table 2 for simpler regarding.

2.8. Nonnegative Tucker Decomposition (NTKD)

The **TKD!** (**TKD!**), also called the Tucker3 or best rank (J, R, P) approximation, can be formulated as follows. Given a third-order data tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ and three positive indices $J_1, J_2, J_3 << I_1, I_2, 1_3$, find a core tensor $\mathcal{G} \in \mathbb{R}^{J_1 \times J_2 \times J_3}$ and three component matrices called factor or loading matrices $\mathbf{U}_1 \in \mathbb{R}^{I_1 \times J_1}$, $\mathbf{U}_2 \in \mathbb{R}^{I_2 \times J_2}$ and $\mathbf{U}_3 \in \mathbb{R}^{I_3 \times J_3}$ which perform the following approximate decomposition:

$$\mathcal{X} \approx \mathcal{G} \times_1 \mathbf{U}^{(1)} \cdots \times_N \mathbf{U}^{(N)} \tag{1}$$

where the core tensor preserves the level of interaction for each factor or projection matrix $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times J_n}$. These matrices are usually, but not necessarily, orthogonal, and can be thought of as the principal components in each mode [17] (see Figure 1). J_n represents the number of components in the decomposition; i.e., the rank $- (R_1, \ldots, R_N)$. We compute rank $- (R_1, \ldots, R_N)$, where rank$_n(\mathcal{X}) = R_n$ for every n-mode, which generally does not exactly reproduce $\mathcal{X}$. Starting from (1), the reconstruction of an approximated tensor can be given by where $\hat{\mathcal{X}}$ is the reconstructed tensor. Then, we can acquire the core tensor $\mathcal{G}$ by the multilinear projection

$$\mathcal{G} = \mathcal{X} \times_1 \mathbf{U}^{(1)\mathrm{T}} \cdots \times_N \mathbf{U}^{(N)\mathrm{T}}, \tag{2}$$

where $\mathbf{U}^{(n)\mathrm{T}}$ denotes the transpose matrix of $\mathbf{U}^{(n)}$ for $n = 1, \ldots, N$. The reconstruction error ξ can be computed as

$$\xi(\hat{\mathcal{X}}) = ||\mathcal{X} - \hat{\mathcal{X}}||_F^2, \tag{3}$$

where $|| \cdot ||_F$ represents the Frobenius norm. To effectively compress data, the reconstructed lower-rank tensor $\hat{\mathcal{X}}$ should be close to the original tensor $\mathcal{X}$; this can be reached by an algorithm as HOOI, which is iterative, and it is described in Section **??**.

$$\hat{\mathcal{X}} = \mathcal{G} \times_1 \mathbf{U}^{(1)} \cdots \times_N \mathbf{U}^{(N)}, \tag{4}$$

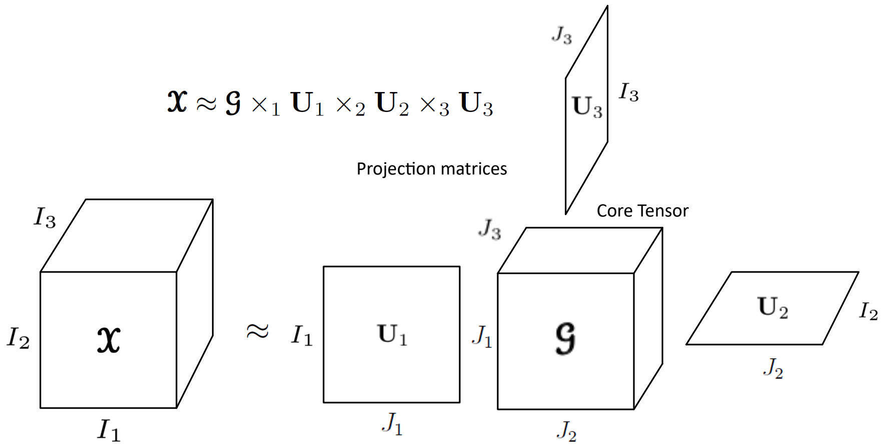

Figure 1. Tucker decomposition for a third-order tensor.

3. Problem phenomenology

3.1. Spectral Imagery

Multi- or Hyper-spectral images are by nature multidimensional nonnegative arrays. A spectral image can be sorted and represented as a thir order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, where $I_!$, I_2 and I_3 represent the height, widht and spectral bands respectively.

...

3.2. Problem Statement

Given a Spectral Image as a third-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, and its corresponding pixelwise classification ground truth matrix $\mathbf{Y} \in \mathbb{R}^{I_1 \times I_2}$ for a specific number of classes C, find, through Non-negative Tensor Factorization, a core tensor $\mathcal{G} \in \mathbb{R}^{J_1 \times J_2 \times J_3}$, where $I_n < J_n$ for $n = 1, \ldots, 3$, to be the input of a natural disaster Detection Neural Network and produce an output matrix $\hat{\mathbf{Y}}$ of predicted classes, achieving higher or competitive performance metrics for pixel-wise classification and decreasing computational complexity in the classification task.

3.3. Mathematical Definition

The problem stated above can be defined mathematically as an optimization problem as follows.

4. Deep Convolutional Neural Networks (DCNNs)

CNNs are supervised feed-forward DL-ANNs for computer vision. The idea of applying a sort of convolution of the synaptic weights of a neural network through the input data yields to a preservation of spatial features, which alleviates the hard task of classification and in turn semantic segmentation. This type of ANN works under the same linear regression model as every machine learning (ML) algorithm. Since images are three dimensional arrays, we can use tensor algebra notation to describe the input of CNNs as a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, where I_1, I_2, and I_3 represent height, width, and depth of the third order array respectively; i.e., the spatial and spectral domain of an image. We can write generally the linear regression model used for ANNs as

$$\hat{\mathbf{y}} = \sigma\left(\mathbf{W}\mathbf{g} + \mathbf{b}\right) \tag{5}$$

where $\hat{\mathbf{y}}$ represents the output prediction of the network; σ denotes an activation function; $\mathbf{g}$ is the input dataset; $\mathbf{W}$ and $\mathbf{b}$ are the matrix of synaptic weights and the bias vector, respectively. These parameters are adjustable; i.e., their values are modified every iteration looking for convergence to minimize the loss in the prediction through optimization algorithms [32]. For simplicity, the bias vector can be ignored, assuming that matrix $\mathbf{W}$ will update until convergence independently of another parameter [32]. Considering that the input dataset to a CNN is a multidimensional array, we can represent (**??**) and (5) using tensor algebra notation as

$$\hat{\mathcal{Y}} = \sigma\left(\mathcal{W}\mathcal{G}\right) \tag{6}$$

where $\hat{\mathcal{Y}}$ represents the prediction output tensor of the ANN (in our case, a second order tensor or matrix $\hat{\mathbf{Y}}$), $\mathcal{G}$ is the input dataset, and $\mathcal{W}$ is a $K_1 \times K_2 \times F_1$ tensor called filter or kernel with the adaptable synaptic weights. Different to conventional ANN, in CNNs, $\mathcal{W}$ is a shiftable square tensor is much smaller in height and width than the input data, i.e., $K_1 = K_2$ and $K_s << I_s$ for $s = 1, 2$; F_1 denotes the number of input channels; i.e., $F_1 = I_3$. For hidden layers, instead of the prediction tensor $\hat{\mathcal{Y}}$, the output is a matrix called activation map $\mathbf{M} \in \mathbb{R}^{I_1 \times I_2}$, which preserves features from the original data in each domain. Actually, it is necessary to use much kernels $\mathcal{W}^{(f_2)}$ as activation maps, with different initialization values to preserve diverse features of the image. Hence, we can also define activation maps as a tensor $\mathcal{M} \in \mathbb{R}^{I_1 \times I_2 \times F_2}$ where F_2 denotes the number of activation maps produced by each filter (see Figure 2). Kernels are displaced through the whole input image as a discrete convolution operation. Then, each element of the output activation map $m_{i_1 i_2 f_2}$ is computed by the summary of the Hadamard product of kernel $\mathcal{W}^{(f_2)}$ and a subtensor from the input tensor $\mathcal{G}$ centered in position (i, j) and with same dimensions of $\mathcal{W}$, as follows

$$m_{i_1 i_2 f_2} = \sigma\left[\sum_{k_1=1}^{K_1}\sum_{k_2=1}^{K_2}\sum_{f_1=1}^{F_1} w_{k_1,k_2,f_1} g_{i_1+k_1-o_1, i_2+k_2-o_2, f_1}\right] \tag{7}$$

where $m_{i_1 i_2 f_2}$ denotes the value of the output activation map f_2 at position i_1, i_2; σ represents the activation function; and o_1 and o_2 are offsets in spatial dimensions which depend on the kernel size, and equal $\frac{K_1+1}{2}$ and $\frac{K_2+1}{2}$ respectively (see Figure 2).

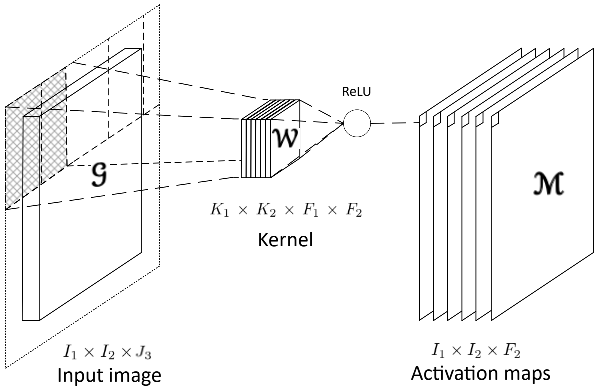
ReLU
$\mathcal{W}$
$\mathcal{G}$
$\mathcal{M}$
$K_1 \times K_2 \times F_1 \times F_2$
Kernel
$I_1 \times I_2 \times J_3$
Input image
$I_1 \times I_2 \times F_2$
Activation maps

Figure 2. Convolutional layer with a $K_1 \times K_2 \times F_1 \times F_2$ kernel. Input channels F_1 must equal the spectral bands I_3. To preserve original dimensions at the output, zero padding is needed [18]. Output dimensions also depend on stride $S = 1$ to consider every piece of pixel information and to preserve original dimensions.

An ANN is trained by using iterative gradient-based optimizers, such as Stochastic gradient descent, Momentum, RMSprop, and Adam [32]. This drive the cost function $L(\mathcal{W})$ to a very low value by updating the synaptic weights $\mathcal{W}$. We can compute the cost function by any function that measures the difference between the training data and the prediction, such as Euclidean distance or cross-entropy [10]. Besides, the same function is used to measure the performance of the model during testing and validation. In order to avoid overfitting [32], the total cost function used to train an ANN combines one of the cost functions mentioned before, plus a regularization term.

$$J(\mathcal{W}) = L(\mathcal{W}) + R(\mathcal{W}), \tag{8}$$

where $J(\mathcal{W})$ denotes the total cost function and $R(\mathcal{W})$ represents a regularization function. Then, we can decrease $J(\mathcal{W})$ by updating the synaptic weights in the direction of the negative gradient. This is known as the method of steepest descent or gradient descent.

$$\mathcal{W}' = \mathcal{W} - \alpha \nabla_{\mathcal{W}} J(\mathcal{W}), \tag{9}$$

where $\mathcal{W}'$ represents the synaptic weights tensor in next iteration during training, α denotes the learning rate parameter, and $\nabla_{\mathcal{W}} J(\mathcal{W})$ the cost function gradient. Gradient descent converges when every element of the gradient is zero, or in practice, very close to zero [10].

CNNs has been successfully used in many image classification frameworks. This variation in architecture from other typical ANN models yields the network to learn spatial and spectral features, which are highly profitable for image classification. Besides, FCNs, constructed with only convolutional layers are able to classify each element of the input image; i.e., they yield pixel-wise classification, or in other words, semantic segmentation.

5. NNTKD for DCNNs

6. Experimental Results

6.1. Input Data

6.1.1. The Training Space

6.1.2. The Labels

6.1.3. The Testing Space

6.1.4. Downloading Data

Code will be delivered by the corresponding author upon request for research purposes only.

6.2. Metrics

7. Discussion and Comparison

8. Conclusions

8.1. Figures, Tables and Schemes

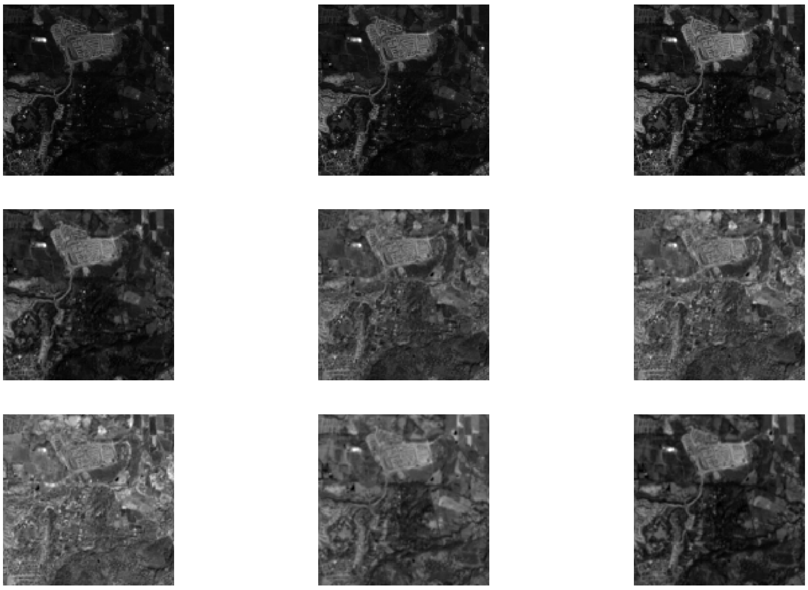

Figure 3. Original Sentinel-2 spectral bands.

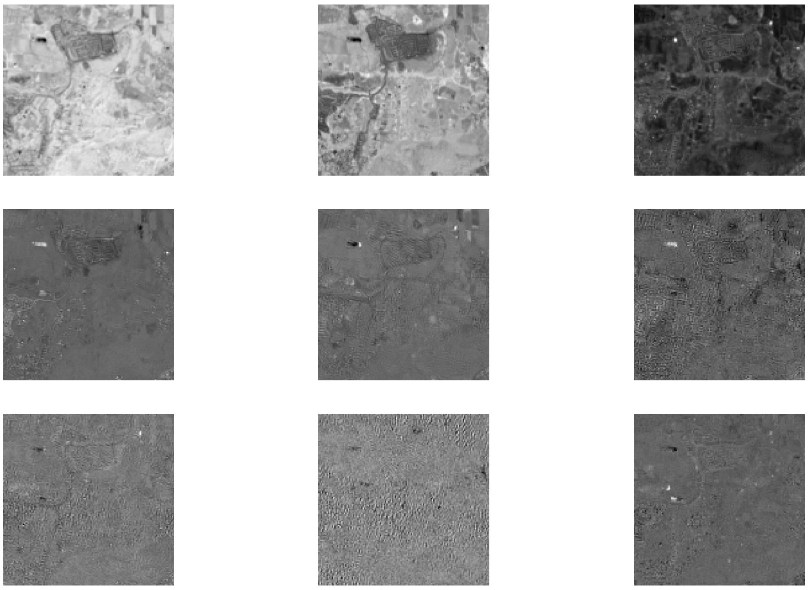

Figure 4. Tucker Decomposition Tensor bands.

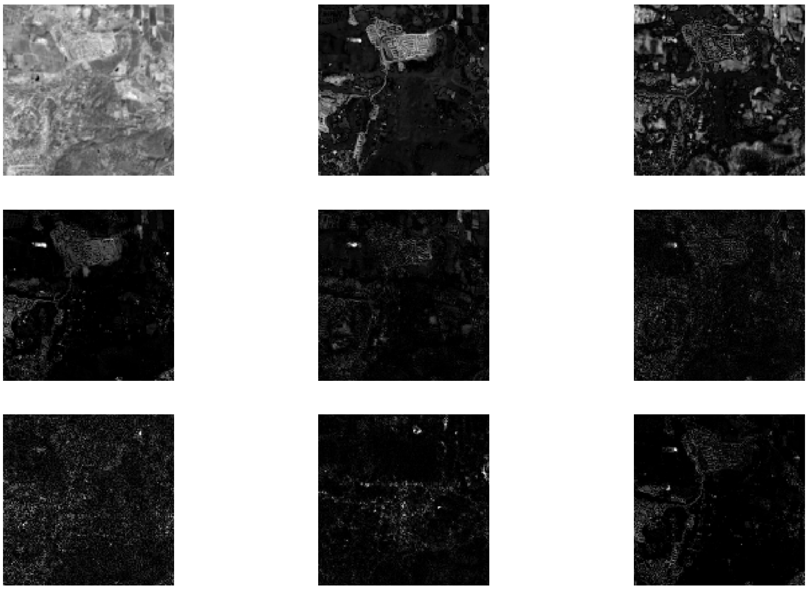

Figure 5. Nonnegative Tucker Decomposition Tensor bands.

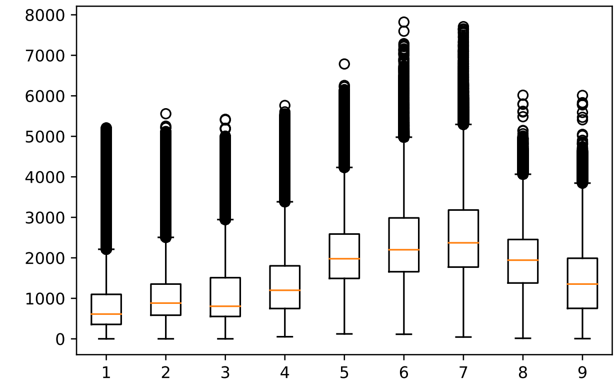

Figure 6. Box and whiskers plot for the original spectral bands.

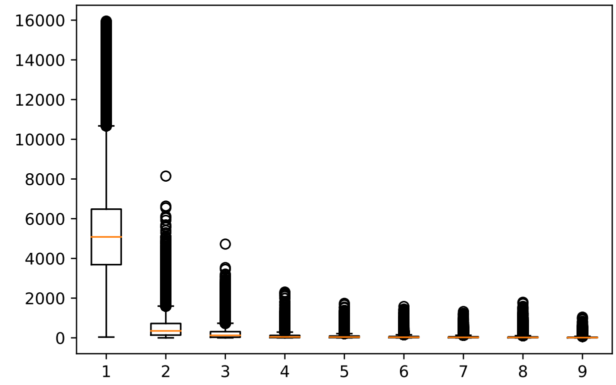

Figure 7. Box and whiskers plot for the original spectral bands.

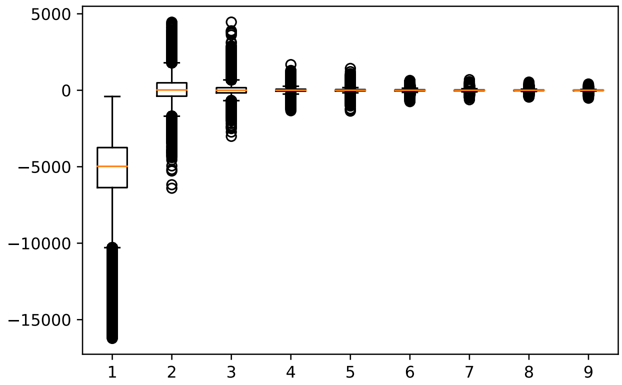

Figure 8. Box and whiskers plot for the original spectral bands.

151 **Author Contributions:** Conceptualization, J.L.; formal analysis, D.T.; investigation, J.L.; methodology, J.L., D.T., and C.A.; resources, C.A.; software, J.L.; supervision, D.T. and C.A.; validation, D.T. and C.A.; writing—original draft, J.L. and D.T.

154 **Funding:** This work was supported by the National Council of Science and Technology CONACYT of Mexico under grant XXXXXXX.

Acknowledgments:

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

Abbreviations

The following abbreviations are used in this manuscript:

ANN	Artificial Neural Network
CNN	Convolutional neural network
CPD	Canonical Polyadic Decomposition
DL	Deep Learning
FCN	Fully Convolutional Network

References

1. Tempfli, K.; Huurneman, G.; Bakker, W.; Janssen, L.; Feringa, W.; Gieske, A.; Grabmaier, K.; Hecker, C.; Horn, J.; Kerle, N.; et al. *Principles of Remote Sensing: An Introductory Textbook*, 4th ed.; ITC: Geneva, Switzerland, 2009.
2. He, Z.; Hu, J.; Wang, Y. Low-rank tensor learning for classification of hyperspectral image with limited labeled sample. *IEEE Signal Process.* **2017**, *145*, 12–25.
3. Richards, A.; Xiuping, J.J. Band selection in sentinel-2 satellite for agriculture applications. In *Remote Sensing Digital Image Analysis*, 4th ed.; Springer-Verlag: Berlin, Germany, 2006.
4. Zhang, T.; Su, J.; Liu, C.; Chen, W.; Liu, H.; Liu, G. Band selection in sentinel-2 satellite for agriculture applications. In Proceedings of the 23rd International Conference on Automation & Computing, University of Huddersfield, Huddersfield, UK, 7–8 September 2017.
5. Xie, Y.; Zhao, X.; Li, L.; Wang, H. Calculating NDVI for Landsat7-ETM data after atmospheric correction using 6S model: A case study in Zhangye city, China. In Proceedings of the 18th International Conference on Geoinformatics, Beijing, China, 18–20 June 2010.
6. Gao, B. NDWI—A normalized difference water index for remote sensing of vegetation liquid water from space. *Remote Sens. Environ.* **1996**, *58*, 1–6.
7. Ham, J.; Chen, Y.; Crawford, M.; Ghosh, J. Investigation of the random forest framework for classification of hyperspectral data. *IEEE Trans. Geosci. Remote Sens.* **2005**, *43*, 492–501.
8. Hearst, Marti A. Support Vector Machines. *IEEE Intell. Syst. J.* **1998**, *13*, 18–28.
9. Huang, X.; Zhang, L. An SVM Ensemble Approach Combining Spectral, Structural, and Semantic Features for the Classification of High-Resolution Remotely Sensed Imagery. *IEEE Trans. Geosci. Remote Sens.* **2013**, *51*, 257–272.
10. Delalieux, S.; Somers, B.; Haest, B.; Spanhove, T.; Vanden Borre, J.; Mucher, S. Heathland conservation status mapping through integration of hyperspectral mixture analysis and decision tree classifiers. *Remote Sens. Environ.* **2012**, *126*, 222—231.
11. Kemker, R.; Salvaggio, C.; Kanan, C. Algorithms for semantic segmentation of multispectral remote sensing imagery using deep learning. *ISPRS J. Photogramm. Remote Sens.* **2018**, *145*, 60–77.
12. Pirotti, F.; Sunar, F.; Piragnolo, M. Benchmark of machine learning methods for classification of a sentinel-2 image. In Proceedings of the XXIII ISPRS Congress, Prague, Czech Republic, 12–19 July 2016.
13. Mateo-García, G.; Gómez-Chova, L.; Camps-Valls, G. Convolutional neural networks for multispectral image cloud masking. In Proceedings of the IGARSS, Fort Worth, TX, USA, 23–28 July 2017.
14. Guo, X.; Huang, X.; Zhang, L.; Zhang, L.; Plaza, A.; Benediktsson, J. A. Support Tensor Machines for Classification of Hyperspectral Remote Sensing Imagery. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 3248–3264.
15. Cichocki, A.; Mandic, D.; De Lathauwer, L.; Zhou, G.; Zhao, Q.; Caiafa, C.; Phan, H. Tensor Decompositions for Signal Processing Applications: From two-way to multiway component analysis. *IEEE Signal Process. Mag.* **2015**, *32*, 145–163.
16. Jolliffe, I.T. *Principal Component Analysis*, 2nd ed.; Springer Verlag: New York, NY, USA, 2002.
17. Kolda, T.; Bader, B. Tensor Decompositions and Applications. *SIAM Rev.* **2009**, *51*, 455–500.

18. Lopez, J.; Santos, S.; Torres, D.; Atzberger, C. Convolutional Neural Networks for Semantic Segmentation of Multispectral Remote Sensing Images. In Proceedings of the LATINCOM, Guadalajara, Mexico, 14–16 November 2018.
19. European Space Agency. Available online: https://sentinel.esa.int/web/sentinel/missions/sentinel-2 (accessed on 15 July 2019).
20. Kemker, R.; Kanan, C. Deep Neural Networks for Semantic Segmentation of Multispectral Remote Sensing Imagery. *arXiv* **2017**, arXiv:abs/1703.06452.
21. Hamida, A.; Benoît, A.; Lambert, P.; Klein, L.; Amar, C.; Audebert, N.; Lefèvre, S. Deep learning for semantic segmentation of remote sensing images with rich spectral content. In Proceedings of the IGARSS, Fort Worth, TX, USA, 23–28 July 2017.
22. Wang, Q.; Lin, J.; Yuan, Y. Salient Band Selection for Hyperspectral Image Classification via Manifold Ranking. *IEEE Trans. Neural Netw. Learn. Syst.* **2016**, *27*, 1279–1289.
23. Li, S.; Qiu, J.; Yang, X.; Liu, H.; Wan, D.; Zhu. Y. A novel approach to hyperspectral band selection based on spectral shape similarity analysis and fast branch and bound search. *Eng. Appl. Artif. Intell.* **2014**, *27*, 241–250.
24. Zhang. L.; Zhang, L.; Tao, D.; Huang, X.; Du, B. Compression of hyperspectral remote sensing images by tensor approach. *Neurocomputing* **2015**, *147*, 358–363.
25. Astrid, M.; Lee, Seung-Ik. CP-decomposition with Tensor Power Method for Convolutional Neural Networks compression. In Proceedings of the BigComp, Jeju, Korea, 13–16 February 2017.
26. Chien, J.; Bao, Y. Tensor-factorized neural networks. *IEEE Trans. Neural Networks Learn. Syst.* **2018**, *29*, 1998–2011.
27. An, J.; Lei, J.; Song, Y.; Zhang, X.; Guo J. Tensor Based Multiscale Low Rank Decomposition for Hyperspectral Images Dimensionality Reductio. *Remote Sens.* **2019**, *11*, 1485.
28. Li, J.; Liu, Z. Multispectral Transforms Using Convolution Neural Networks for Remote Sensing Multispectral Image Compression. *Remote Sens.* **2019**, *11*, 759.
29. An, J.; Song, Y.; Guo, Y.; Ma, X.; Zhang, X. Tensor Discriminant Analysis via Compact Feature Representation for Hyperspectral Images Dimensionality Reduction. *Remote Sens.* **2019**, *11*, 1822.
30. Absil, P.-A.; Mahony, R.; Sepulchre, R. *Optimization Algorithms on Matrix Manifolds*, 1st ed.; Princeton University Press: Princeton, NJ, USA, 2007.
31. De Lathauwer, L.; De Moor, B.; Vandewalle, J. On the best rank-1 and rank-(R 1 ,R 2 ,···,R N) approximation of higher-order tensors. *SIAM J. Matrix Anal. Appl.* **2000**, *21*, 1324–1342.
32. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*, 1st ed.; MIT Press, 2016.
33. Sheehan, B. N.; Saad, Y. Higher Order Orthogonal Iteration of Tensors (HOOI) and its Relation to PCA and GLRAM. In Proceedings of the 7th SIAM International Conference on Data Mining, Minneapolis, MN, USA, 26–28 April 2007.
34. Badrinarayanan, V.; Kendall, A.; Cipolla, R. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 2481–2495.
35. De Lathauwer, L.; De Moor, B.; Vandewalle, J. A Multilinear Singular Value Decomposition. *SIAM J. Matrix Anal. Appl.* **2000**, *21*, 1253–1278.
36. Rodes, I.; Inglada, J.; Hagolle, O.; Dejoux, J.; Dedieu, G. Sampling strategies for unsupervised classification of multitemporal high resolution optical images over very large areas. In Proceedings of the 2012 IEEE International Geoscience and Remote Sensing Symposium, Munich, Germany, 22–27 July 2012.

Sample Availability: Samples of the compounds ...... are available from the authors.