

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 22, 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 which 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 recent years [1]. The insertion of tensor-based algorithms for this sort of tasks drove a revolution in several areas such as image processing [2].

Most of the researches in the image processing area require data acquired by multiple sensors. Even in the simplest case, color images, data acquired by three sensors that perceive reflectance of an object are processed. Each sensor receives reflectance in different wavelength ranges, and by merging that data, a color image is produced. Thus, the images can be represented in a form of three-dimensional arrays or third-order tensors. Two dimensions represent the spatial properties and the third dimension denotes the "depth", or rather the spectral bands. Medical analysis [3], mineralogy [4], agriculture, [5], radar images [6] and, above all, remote sensing multi- and hyper-spectral images [7] work, by nature, with third-order tensors.

Spectral imagery remarkably aids certain image processing tasks [8]. Recently the use of this type of data has grown exponentially in various areas [9]. The ability to obtain information about a target not only by its reflectance in the spatial domain, but also by response at different wavelengths, has driven a growth in accuracy and precision in tasks such as classification and segmentation [10].

Initially, several unsupervised classification and segmentation algorithms [11] were developed, taking advantage of the properties that spectral data produce. Subsequently, with the introduction of

32 supervised machine learning algorithms such as SVM [], kNN [] and ANN [], it was found that, under
33 certain conditions, there is a direct relation between the number of bands used and the performance of
34 these algorithms []. However, with the aim of improving results, neural network models evolved into
35 deep neural networks []. This caused that spectral image processing was not productive at all, since
36 the processing time increased considerably. The foregoing requires having robust computer equipment
37 to achieve competitive results in time.

38 Several works have opted for matrix factorization algorithm to reduce the high-dimensionality of
39 spectral images []. More recently, with the development of tensor factorization algorithms [], it has
40 been found that some algorithms based on tensor algebra produce certain advantages over those based
41 on matrices []. Nevertheless, the data produced by both of them are hard to understand for supervised
42 classification algorithms that need spatial relation between pixels to produce a wise prediction [].

43 In this work we propose an alternative solution to the problem described previously. To reduce
44 processing times in supervised classification algorithms, such as deep neural networks, we propose a
45 model that, from the benefits offered by tensor decomposition algorithms, aids compression of spectral
46 images. This produces a lower dimensional tensor while preserving the structural and numerical
47 nature of the original data.

48 1.1. State of art

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

54 Before the introduction of tensor decomposition algorithms, the way to use hyperspectral images
55 as input for supervised classification algorithms was by band selection [23] and [22]. Later, matrix
56 decomposition algorithms were used, such as PCA in [?], and even non-negative matrix decomposition
57 methods [?]. In 2015 Zhang et al. [24] were pioneers in experimenting with multilinear algebra-based
58 decompositions on hyperspectral images.

59 On the other hand, there was also the possibility of using multispectral images due to the small
60 number of spectral bands, which still made efficient results in classification without dimensionality
61 reduction achievable, as done in [11], [18], [21] and [?]. However, the need to increase classification
62 accuracy results forces researchers to use data with greater features that favor and aid the classification
63 of various classes, which are difficult to differentiate with little spectral data. Thus, more recent
64 researches have decided to use hyperspectral images with tensor decompositions, which has increased
65 the results in classification accuracy [26], [27], [29], [?] and [?].

66 Recently, Sayeh et al. [?] published a work close to our research. They proposed a non-negative
67 tensor decomposition of hyperspectral images but, different to our research, they try to preserve certain
68 spatial-spectral features into the so called abundance maps, i.e. the projection matrices, while this work
69 pursues to preserve the nature of the image just compressing the main information in the positive core
70 tensor.

71 Table 1 summarizes some of the most cited related papers, which deal with the
72 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
Kemker, R. et al. [11] (2017)	MSI	-	-	CNN
Tong L. et al. [] (2017)	HSI	NMF	Unmixing	-
Hamida, A. et al. [21] (2017)	MSI	-	-	CNN
Chien, J. et al. [] (2017)	RGB	TFNN	Spatial-Spectral	TFNN
Dewa, M. et al. [] (2018)	HSI	PCA	Spectral	PCA
Xu, Y. et al. [] (2018)	HSI	-	-	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
Lopez, J. et al. [] (2019)	MSI	TKD	Spectral	FCN
Sayeh, M. et al. [] (2019)	HSI	NTD	Spatial-Spectral	3D-CNN
Our framework	MSI/HSI	NTKD	Spectral	CNN

73 1.2. Contribution

74 In the state of art we can find a variety of frameworks looking for dimensionality reduction of
 75 images of any type, that is, RGB [?], medical [?], SAR [?], multispectral [?], among others. In
 76 particular, the large number of spectral bands in hyperspectral images has focused efforts in this
 77 category [?]. Spatial [?], spectral [?] and spatial-spectral [?] compression have been achieved with
 78 matrix and tensor decompositions. It is important to highlight that, a decomposition as pre-processing
 79 for a classification algorithm, instead of favoring, could be counterproductive and greatly affect its
 80 performance. It is therefore important to make a proper selection of the type of decomposition that
 81 would produce a suitable data set for post-processing.

82 Unlike previous works, this work seeks to adapt the data in the best way to be the input of deep
 83 convolutional networks. Convolutional network models are designed to extract and interpret all the
 84 spatial properties of an image by moving the kernels over the input data []. Therefore, producing
 85 uncorrelated data in space and spectrum, would make harder the interpretation of the data in the
 86 convolutional network [?]. Thus, the proposed framework maintains the positive tensor nature of
 87 the spectral images and the spatial dimensionality to preserver spatial-spectral correlation of the data
 88 while reducing the spectral dimensionality, in order to decrease computational load in the pixel-wise
 89 classification process.

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

- 91 1. The framework NTKD3-CNN proposed in this work, develops a new strategy to improve
 92 accuracy results of semantic segmentation convolutional neural networks by finding suitable
 93 tensor data, preserving spatial correlation and values in the set of the natural numbers while
 94 compressing the spectral domain and in turn decreasing computational load.
- 95 2. This work also presents an exhaustive performance analysis measuring and comparing its
 96 efficiency with the most popular metrics, i.e., as pixel accuracy (PA), also PA in function of the
 97 number of new tensor bands, precision, recall, F1, orthogonality degree of the factor matrices
 98 and the core tensor, reconstruction error of the original tensor, and execution time.

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

106 2. Tensor-Based Factorizations

107 Matrix-based factorizations, such as PCA [] and SVD [] have been significant and useful tools for
 108 dimensionality reduction and other approaches. Nevertheless, they are limited by representations
 109 of data only in two modes. Most of current applications have data structures often as higher-order
 110 arrays, e.g. dimensions of space, time, and frequency. This 2-way view in matrix factorizations may be
 111 inadequate and it is natural to use tensor decomposition approaches [?].

112 We can define a tensor as a multi-way or multidimensional array. The order of a tensor is the
 113 number of dimensions, also known as modes, i.e., an N -order tensor $\mathbf{X} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ is an N -dimesional
 114 array, which elements x_{i_1, i_2, \dots, i_N} are indexed by $i_n \in 1, 2, \dots, I_n$ for $1 \leq n \leq N$.

115 Throughout this paper, the mathematical notation used by Kolda et al. [17] has been adopted.
 116 Table 2 summarize this notation.

Table 2. Tensor algebra notation summary

$\mathbf{A}, \mathbf{A}, \mathbf{a}, a$	Tensor, matrix, vector and scalar respectively
$\mathbf{A} \in \mathbb{R}^{I_1 \times \dots \times I_N}$	N -order tensor of size $I_1 \times \dots \times I_N$.
$a_{i_1 \dots i_N}$	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. $\mathbf{A}_{(n)} \in \mathbb{R}^{I_n \times \prod_{m \neq n} I_m}$
$\mathbf{X} = \mathbf{a}^{(1)} \circ \dots \circ \mathbf{a}^{(N)}$	Outer product of N vectors, where $x_{i_1 i_2 \dots i_N} = a_{i_1}^{(1)} \dots a_{i_N}^{(N)}$
$\langle \mathbf{A}, \mathbf{B} \rangle$	Inner product of two tensors.
$\mathbf{B} = \mathbf{A} \times_n \mathbf{U}$	n -mode product of tensor $\mathbf{A} \in \mathbb{R}^{I_1 \times \dots \times I_N}$ by a matrix $\mathbf{U} \in \mathbb{R}^{J \times I_n}$ along axis n .

117 2.1. Basic concepts

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

120 2.1.1. Matricization

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

123 2.1.2. Outer Product

124 The outer product of N vectors $\mathbf{X} = \mathbf{a}^{(1)} \circ \dots \circ \mathbf{a}^{(N)}$ produces a tensor $\mathbf{X} \in \mathbb{R}^{I_1 \times \dots \times I_N}$
 125 where \circ denotes the outer product and $\mathbf{a}^{(n)}$ denotes a vector in a sequence of N vectors
 126 and each element of the tensor is the product of the corresponding vector elements; i.e.,
 127 $x_{i_1 i_2 \dots i_N} = a_{i_1}^{(1)} \dots a_{i_N}^{(N)}$.

128 2.1.3. Inner Product

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

131 2.1.4. N -Mode Product

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

134 2.1.5. Rank-One Tensor

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

¹³⁷ 2.1.6. Rank-R Tensor

¹³⁸ The rank of a tensor \mathfrak{X} is the smallest number of components in a CPD; i.e., the smallest
¹³⁹ number of rank-one tensors that generate \mathfrak{X} as their sum [17].

¹⁴⁰ 2.1.7. N-Rank

¹⁴¹ The n -rank of a tensor $\mathfrak{X} \in \mathbb{R}^{I_1 \times \dots \times I_N}$ denoted $\text{rank}_n(\mathfrak{X})$, is the column rank of $\mathbf{X}_{(n)}$; i.e., the
¹⁴² dimension of the vector space spanned by the mode- n fibers. Hence, if $R_n \equiv \text{rank}_n(\mathfrak{X})$ for $n = 1, \dots, N$,
¹⁴³ we can say that \mathfrak{X} has a rank – (R_1, \dots, R_N) tensor.

¹⁴⁴ 2.2. Nonnegative Tucker Decomposition (NTKD)

¹⁴⁵ The TKD, also called the Tucker3 for the particular case of third-order tensors, or best rank $(J_1,$
¹⁴⁶ $J_2, J_3)$ approximation, can be formally formulated as follows [?]. Given a third-order data tensor
¹⁴⁷ $\mathfrak{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ and three positive indices $J_1, J_2, J_3 << I_1, I_2, I_3$, find a core tensor $\mathfrak{G} \in \mathbb{R}^{J_1 \times J_2 \times J_3}$ and
¹⁴⁸ 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 $\mathbf{U}_3 \in \mathbb{R}^{I_3 \times J_3}$
¹⁴⁹ which perform the following approximate decomposition:

$$\mathfrak{X} = \mathfrak{G} \times_1 \mathbf{U}^{(1)} \dots \times_N \mathbf{U}^{(N)} + \mathcal{E} \quad (1)$$

¹⁵⁰ where \mathcal{E} denotes the approximation error. The core tensor \mathfrak{G} preserves the level of interaction for each
¹⁵¹ factor or projection matrix $\mathbf{U}^{(n)}$. The factor matrices are commonly considered orthogonal, but in
¹⁵² Tucker models with non-negativity constraints, that is not necessarily imposed [?]. These matrices
¹⁵³ can be seen as the principal components in each mode [17] (see Figure 1). J_n represents the number of
¹⁵⁴ components in the decomposition; i.e., the rank – (R_1, \dots, R_N) .

¹⁵⁵ We can also denote the TKD using the matricization approach and express it by

$$\mathbf{X}_{(1)} = \mathbf{U}^{(1)} \mathbf{G}_{(1)} (\mathbf{U}^{(3)} \otimes \mathbf{U}^{(2)})^T \quad (2a)$$

$$\mathbf{X}_{(2)} = \mathbf{U}^{(2)} \mathbf{G}_{(2)} (\mathbf{U}^{(3)} \otimes \mathbf{U}^{(1)})^T \quad (2b)$$

$$\mathbf{X}_{(3)} = \mathbf{U}^{(3)} \mathbf{G}_{(3)} (\mathbf{U}^{(2)} \otimes \mathbf{U}^{(1)})^T \quad (2c)$$

¹⁵⁶ where \otimes denotes the Kronecker product and $\mathbf{X}_{(n)}$ and $\mathbf{G}_{(n)}$ are the n -mode matricized versions of
¹⁵⁷ tensor \mathfrak{X} and \mathfrak{G} respectively.

¹⁵⁸ Starting from (1), the reconstruction of an approximated tensor can be given by

$$\hat{\mathfrak{X}} = \mathfrak{G} \times_1 \mathbf{U}^{(1)} \dots \times_N \mathbf{U}^{(N)} \quad (3)$$

¹⁵⁹ where $\hat{\mathfrak{X}}$ is the reconstructed tensor.

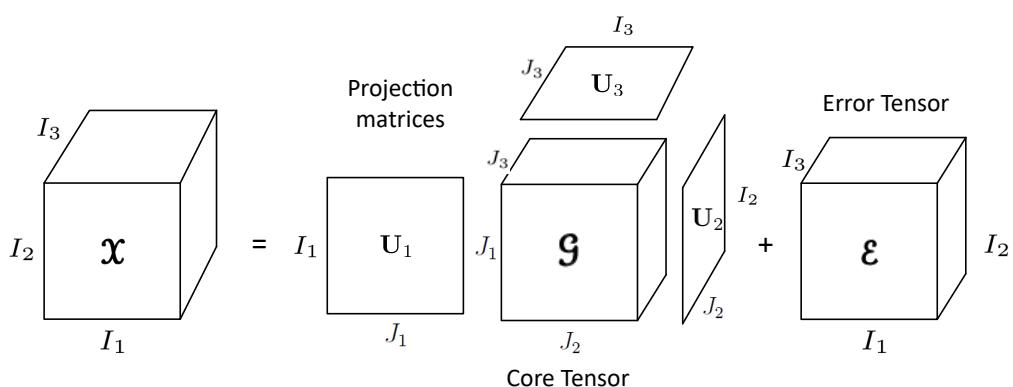


Figure 1. Tucker decomposition for a third-order tensor.

160 Then, we can acquire the core tensor \mathbf{G} by the multilinear projection

$$\mathbf{G} = \mathbf{X} \times_1 \mathbf{U}^{(1)\top} \cdots \times_N \mathbf{U}^{(N)\top} \quad (4)$$

161 where $\mathbf{U}^{(n)\top}$ denotes the transpose matrix of $\mathbf{U}^{(n)}$ for
 162 $n = 1, \dots, N$. The reconstruction error ξ can be computed as

$$\xi(\hat{\mathbf{X}}) = \|\mathbf{X} - \hat{\mathbf{X}}\|_F^2 \quad (5)$$

163 where $\|\cdot\|_F$ represents the Frobenius norm. To compute the best rank approximation of a tensor, it is
 164 required an iterative algorithm. HOSVD, ALS and HOOI are algorithm commonly used in TKD [?].

165 HOOI initializes the factors matrices using HOSVD and assumes that orthogonal matrices are
 166 known, so that the core tensor is obtained with (4). Then, it maximizes the cost function

$$\max_{\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \mathbf{U}^{(3)}} \|\mathbf{X} \times_1 \mathbf{U}^{(1)\top} \times_2 \mathbf{U}^{(2)\top} \cdots \times_N \mathbf{U}^{(N)\top}\|_F^2 \quad (6)$$

167 with $\mathbf{U}^{(n)}$ unknown. Fixing all factor matrices but one, tensor \mathbf{X} can be projected onto the
 168 $\{R_1, \dots, R_{n-1}, R_{n+1}, \dots, R_N\}$ -dimensional space as

$$\mathbf{W}^{(-n)} = \mathbf{X} \times_1 \mathbf{U}^{(1)\top} \cdots \times_{n-1} \mathbf{U}^{(n-1)\top} \times_{n+1} \mathbf{U}^{(n+1)\top} \cdots \times_N \mathbf{U}^{(N)\top} \quad (7)$$

169 and the orthogonal matrices can be estimated as an orthonormal basis for the dominant subspace
 170 of the projection by applying the standard matrix SVD for mode- n unfolded matrix $\mathbf{W}_{(n)}^{(-n)}$ [?]. See
 171 Algorithm 1.

172 The HOOI algorithm evidently can be applied for the particular case of third-order tensor.
 173 Furthermore, it can be slightly adjusted for preserving any input dimension and develop the
 174 decomposition in a specific order. This will be explained in next chapters.

Algorithm 1: HOOI algorithm to compute a rank- (R_1, \dots, R_N) TKD for an N th-order tensor
 $\mathbf{X} \in \mathbb{R}^{I_1 \times \dots \times I_N}$.

Function HOOI($\mathbf{X}, J_1, \dots, J_N$):

 initialize $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times J_n}$ for $n = 1, \dots, N$ using HOSVD or random

repeat

for $n = 1, \dots, N$ **do**

$\mathbf{W}^{(-n)} \leftarrow \mathbf{X} \times_{-n} \{\mathbf{U}^\top\}$

$[\mathbf{U}^{(n)}, \Sigma^{(n)}, \mathbf{V}^{(n)}] \leftarrow \text{svds}(\mathbf{W}_{(n)}^{(-n)}, J_n, 'LM')$

$\mathbf{U}^{(n)} \leftarrow [\mathbf{U}^{(n)}]_+$

end

until fit ceases to improve or maximum iterations exhausted;

$\mathbf{G} \leftarrow \mathbf{W}^{(-N)} \times_N \mathbf{U}^{(N)\top}$

Output: $\mathbf{U}^{(n)} \in \mathbb{R}_+^{I_n \times J_n}$, $\mathbf{G} \in \mathbb{R}_+^{J_1 \times J_2 \times \dots \times J_N}$

177 3. Problem phenomenology

178 3.1. Spectral Imagery

179 Multi- or Hyper-spectral images are by nature multidimensional nonnegative arrays. A spectral
 180 image can be sorted and represented as a third-order tensor $\mathbf{X} \in \mathbb{R}_+^{I_1 \times I_2 \times I_3}$, where I_1, I_2 and I_3 represent
 181 its height, width and spectral bands respectively. In RS image processing, spectral images are frequently
 182 used for classification of different material in a scene of interest. However, due to the low spatial

¹⁸³ resolution produced by the distance between the sensor and the target, spatial features are not sufficient
¹⁸⁴ to discern certain classes. That is why spectral resolution plays an important role in this type of task.

¹⁸⁵ The separation into spectral bands allows perception of reflectance at different wavelengths. This
¹⁸⁶ helps to better characterize various materials, in order to simplify the process of discernment between
¹⁸⁷ classes. The effort to obtain these spectral features generates a greater amount of data, which increases
¹⁸⁸ the processing complexity. This is where the spectral decomposition task becomes relevant.

¹⁸⁹ 3.2. Problem Statement

¹⁹⁰ Given a Spectral Image as a third-order tensor $\mathbf{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, and its corresponding pixelwise
¹⁹¹ classification ground truth matrix $\mathbf{Y} \in \mathbb{R}^{I_1 \times I_2}$ for a specific number of classes C , find, through
¹⁹² Non-negative Tensor Decomposition, the best rank- (R_1, R_2, R_n) approximation and its respective
¹⁹³ core tensor $\mathbf{G} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, where $R_1 = I_1, R_2 = I_2$ and $R_3 = J_3 << I_3$, to be the input of a pixel-wise
¹⁹⁴ classification Convolutional Neural Network and produce an output matrix $\hat{\mathbf{Y}}$ of predicted classes,
¹⁹⁵ achieving competitive performance metrics for pixel-wise classification while decreasing computational
¹⁹⁶ load in the classification process.

¹⁹⁷ 3.3. Mathematical Definition

¹⁹⁸ We can mathematically define the problem statement described above as an optimization problem.

$$\begin{aligned} & \min_{\mathbf{G}, \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \mathbf{U}^{(3)}} \|\mathbf{X} - \mathbf{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{U}^{(3)}\|_F^2 \\ & \text{subject to } \mathbf{U}^{(n)} \in \mathbb{R}_+^{I_n \times J_n} \quad \text{for } n = 1, 2, 3 \quad \text{and} \quad \mathbf{G} \in \mathbb{R}_+^{I_1 \times I_2 \times J_3} \\ & \quad J_1 = I_1, J_2 = I_2 \quad \text{no compression in the spatial domain,} \\ & \quad J_3 << I_3 \quad \text{reduced spectral domain at the core tensor,} \\ & \quad \xi(\hat{\mathbf{X}}) \rightarrow 0 \quad \text{and} \quad \text{PA} \geq \psi \quad \text{competitive pixel accuracy up to a threshold} \end{aligned} \tag{8}$$

¹⁹⁹ 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 $\mathbf{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, 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

$$\hat{\mathbf{y}} = \sigma(\mathbf{W}\mathbf{g} + \mathbf{b}) \tag{9}$$

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 (9) using tensor algebra notation as

$$\hat{\mathbf{y}} = \sigma(\mathbf{W}\mathbf{G}) \tag{10}$$

where $\hat{\mathbf{y}}$ represents the prediction output tensor of the ANN (in our case, a second order tensor or matrix $\hat{\mathbf{Y}}$), \mathbf{G} is the input dataset, and \mathbf{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, \mathbf{W} is a shiftable square tensor is much smaller in height and width than the input data, i.e., $K_1 = K_2$ and $K_s \ll I_s$ 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 $\hat{\mathbf{y}}$, 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 $\mathbf{W}^{(f_2)}$ 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 $\mathbf{W}^{(f_2)}$ and a subtensor from the input tensor \mathbf{G} centered in position (i, j) and with same dimensions of \mathbf{W} , as follows

$$m_{i_1 i_2 f_2} = \sigma \left[\sum_{k_1=1}^{K_1} \sum_{k_2=1}^{K_2} \sum_{f_1=1}^{F_1} w_{k_1, k_2, f_1} g_{i_1+k_1-o_1, i_2+k_2-o_2, f_1} \right] \quad (11)$$

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).

203

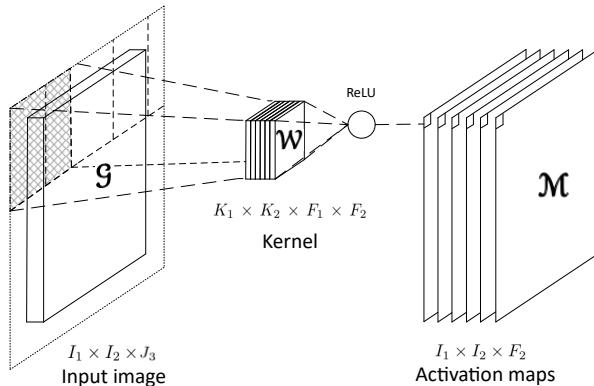


Figure 2. Convolutional layer with a $K_1 \times K_2 \times F_1 \times F_2$ kernel. Input channels F_1 must equal the spectral bands I_3 . To preserve original dimensions at the output, zero padding is needed [18]. Output dimensions also depend on stride $S = 1$ to consider every piece of pixel information 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 $L(\mathbf{W})$ to a very low value by updating the synaptic weights \mathbf{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.

$$J(\mathbf{W}) = L(\mathbf{W}) + R(\mathbf{W}), \quad (12)$$

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

$$\mathbf{W}' = \mathbf{W} - \alpha \nabla_{\mathbf{W}} J(\mathbf{W}), \quad (13)$$

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

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

212 5. NNTKD for DCNNs

213 6. Experimental Results

214 6.1. Input Data

215 For this work, we chose three of the most popular multi- and hyperspectral dataset for
 216 classification.

217 6.1.1. Sentinel-2 CNNMSI

218 This dataset developed by Lopez et al. [?] is composed of RS Sentinel-2 scenarios from central
 219 Europe. It has 100 scenarios for the training space and 10 scenarios for testing, all of them with
 220 128×128 pixels with spatial resolution of $20m^2$ and 9 spectral bands in the range $490 - 2190nm$.
 221 The labels are semi-manually assigned for five classes of interest: vegetation, soil, water, clouds and
 222 shadows. Data are available in the link [Sentinel-2 Dataset](#).

223 6.1.2. The Training Space Indian Pines

224 This dataset is a scene produced by AVIRIS in North-western Indiana and consists of 145×145
 225 pixels and 224 spectral bands in the wavelength range $0.4'2.5\mu m$. The Indian Pines scene contains
 226 two-thirds agriculture, and one-third forest or other natural perennial vegetation. There are two
 227 major dual lane highways, a rail line, as well as some low density housing, other built structures, and
 228 smaller roads. Since the scene is taken in June some of the crops present, corn, soybeans, are in early
 229 stages of growth with less than 5% coverage. The ground truth available is designated into sixteen
 230 classes and is not all mutually exclusive. Indian Pines data are available at [Indian Pines dataset](#).

231 6.1.3. The Labels Salinas

232 This scene was collected by the AVIRIS sensor over Salinas Valley, California. It has 512×217
 233 pixels with spatial resolution $3.7m$, and 224 spectral bands. It includes vegetables, bare soils, and
 234 vineyard fields. Salinas groundtruth contains 16 classes.

²³⁵ 6.1.4. **The Testing Space**

Table 3. Summary of the different dataset used for experiments in this work.

Dataset	Spatial dimensions	Bands	Classes	Samples
Sentinel-2 CNNMSI	128×128	5	9	147,456
Indian Pines	145×145	224	16	4,709,600
Salinas	512×217	224	16	24,887,296

²³⁶ 6.2. **Metrics**

²³⁷ 6.2.1. **Downloading Data** Relative Mean Square Error (rMSE)

²³⁸ In order to compute the reconstruction error of the tensor $\mathbf{\hat{x}}$ for the implementation of HOOL,
²³⁹ the rMSE was used:

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

$$rMSE(\mathbf{\hat{x}}) = \frac{1}{Q} \sum_{q=1}^Q \frac{\|\mathbf{\hat{x}}_q - \mathbf{x}_q\|_F^2}{\|\mathbf{x}_q\|_F^2}, \quad (14)$$

²⁴⁰ where \mathbf{x}_q represents the q -th CNNMSI from our dataset with Q MSIs and $\mathbf{\hat{x}}_q$ its corresponding
²⁴¹ reconstruction computed by (3).

²⁴² 6.3. **Metrics**

²⁴³ 6.2.1. **Cohen's Kappa Coefficient**

²⁴⁴ Cohen's kappa coefficient is a very robust metric used to measure reliability of multi-class and
²⁴⁵ imbalanced class classification algorithms. It is computed by

$$\kappa = \frac{\rho_o - \rho_e}{1 - \rho_e} \quad (15)$$

²⁴⁶ where ρ_o is the observed agreement, and ρ_e is the expected agreement. This metric will always
²⁴⁷ produce values less than or equal to 1, where 1 means perfect agreement. Negative values indicate
²⁴⁸ no agreement, i.e., futile classification.

²⁴⁹ 6.2.2. **Pixel Accuracy (PA)**

²⁵⁰ We used the PA metric to compute a ratio between the amount of correctly classified pixels and
²⁵¹ the total number of pixels as follows. Given a confusion matrix relating the True Positive (TP), True
²⁵² Negatives (TN), False Positives (FP) and False Negatives (FN), the PA is computed by

$$PA = \frac{\sum_{c=1}^C \tau_{cc}}{\sum_{c=1}^C \sum_{d=1}^C \tau_{cd}} = \frac{TP + TN}{TP + TN + FP + FN} \quad (16)$$

²⁵³ where c is the number of class for $c = 1, \dots, C$ and τ_{cc} is the amount of pixels of class c correctly
²⁵⁴ assigned to class c i.e., TP and TN, and τ_{cd} is the amount of pixels of class c inferred to belong to class
²⁵⁵ d . Despite this metric is wide used, it is not a totally fair metric for imbalanced classes. In this work
²⁵⁶ we used this metric with comparison purposes. However, we also used metrics more in line with
²⁵⁷ multi class classification tasks.

²⁵⁸ 6.2.3. Precision

²⁵⁹ Another metric used in this work as a performance evaluation metric in multiclass classification
²⁶⁰ is precision. This is a metric that measures the percentage of pixels from class c correctly classified. It
²⁶¹ is computed with the following equation

$$\Psi_c = \frac{\tau_{cc}}{\sum_{d=1}^C \tau_{cd}} = \frac{TP}{TP + FP} \quad (17)$$

²⁶² where Ψ_c denotes the precision of class c , which is the number of TP divided by the TP plus FP.

²⁶³ 6.2.4. Recall or Sensitivity

²⁶⁴ Recall, or also known as sensitivity is a metric that indicates the proportion of pixels classified as
²⁶⁵ class c that actually belong to class c . It is computed with the following equation

$$v_c = \frac{\tau_{cc}}{\sum_{d=1}^C \tau_{dc}} = \frac{TP}{TP + FN} \quad (18)$$

²⁶⁶ where v_c denotes the recall of class c for $c = 1, \dots, C$.

²⁶⁷ 6.2.5. F1 Score

²⁶⁸ In order to summarize precision and recall in one only metric, we use the F1 score, which is
²⁶⁹ computed by

$$F1 = \frac{2\Psi v}{\Psi + v} \quad (19)$$

²⁷⁰ This metric provides a very appropriate measure of multiclass classification for imbalanced dataset.

²⁷¹ 7. Discussion and Comparison

²⁷² 8. Conclusions

²⁷³ 8.1. *Figures, Tables and Schemes*

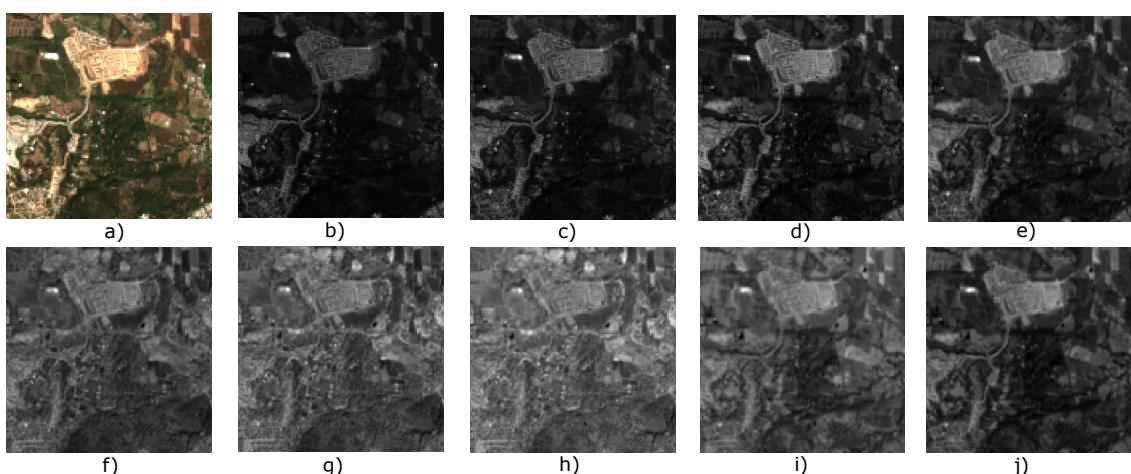


Figure 3. Original Sentinel-2 spectral bands, a) True color image, b) to j) 1st to 9th spectral band respectively.

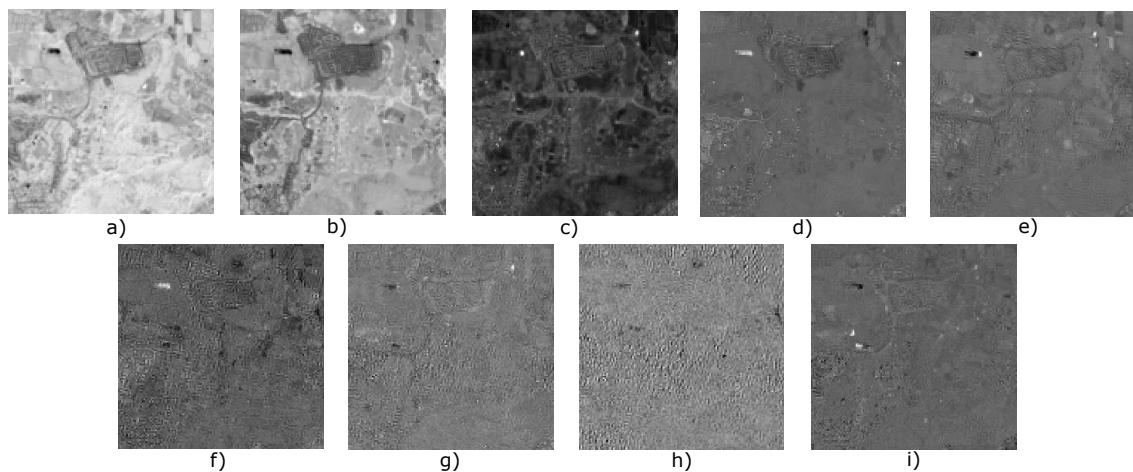


Figure 4. Tensor bands from the Tucker Decomposition, a) to i) 1st to 9th band respectively.

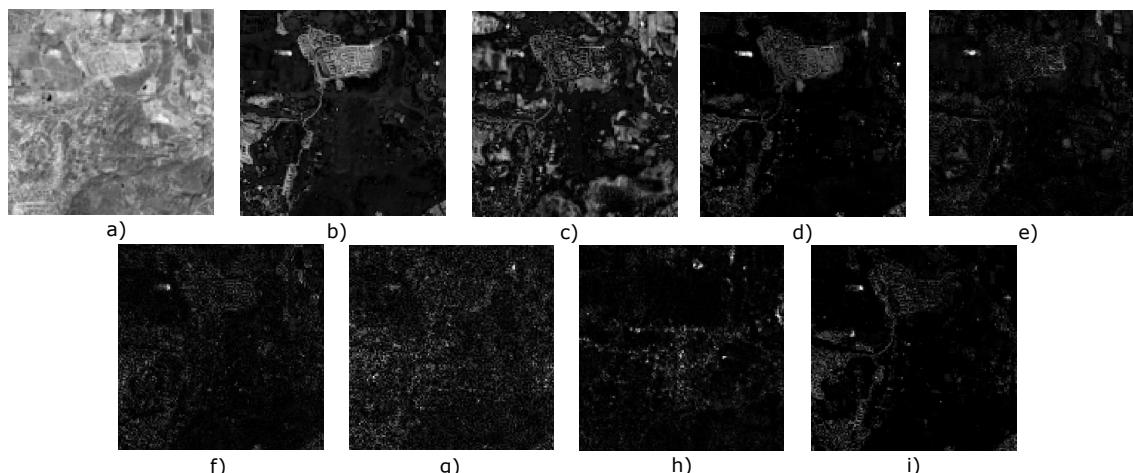


Figure 5. Tensor bands from the Non-negative Tucker Decomposition, a) to i) 1st to 9th band respectively.

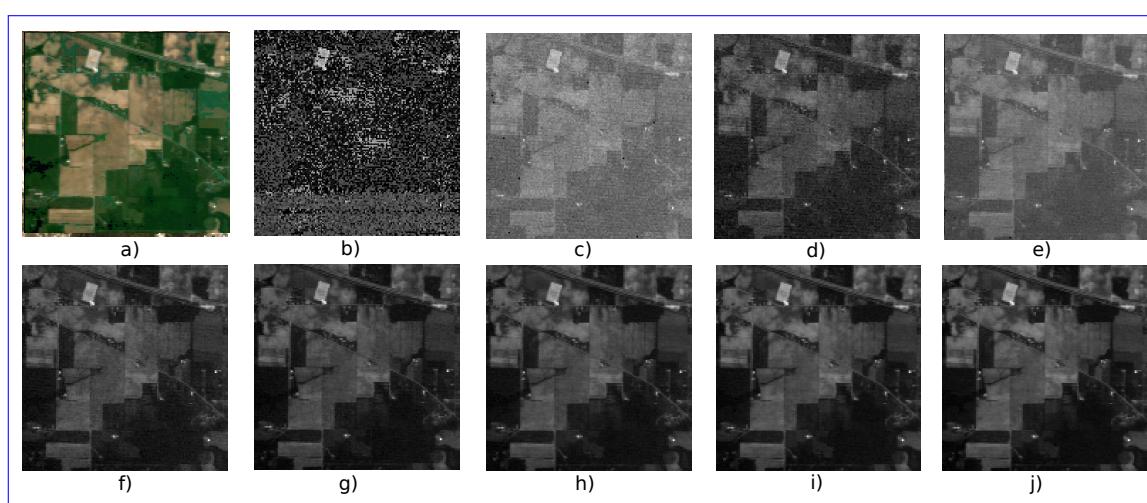


Figure 6. Original Sentinel-2 spectral bands.

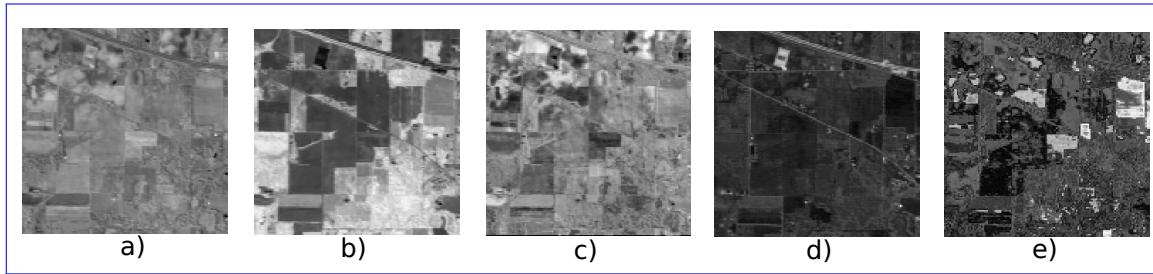


Figure 7. Tucker Decomposition Tensor bands.

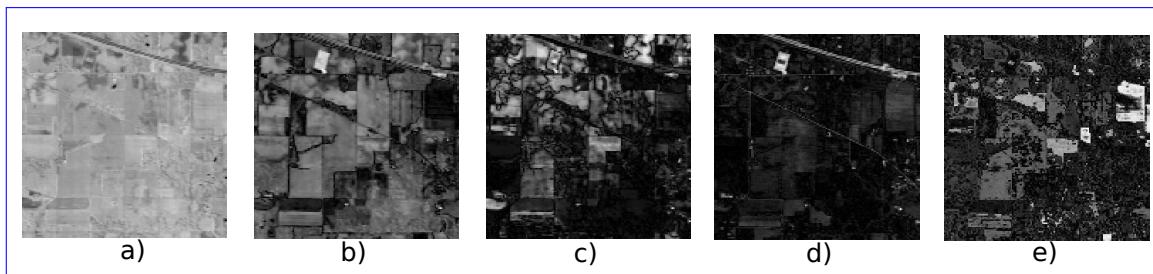


Figure 8. Nonnegative Tucker Decomposition Tensor bands.

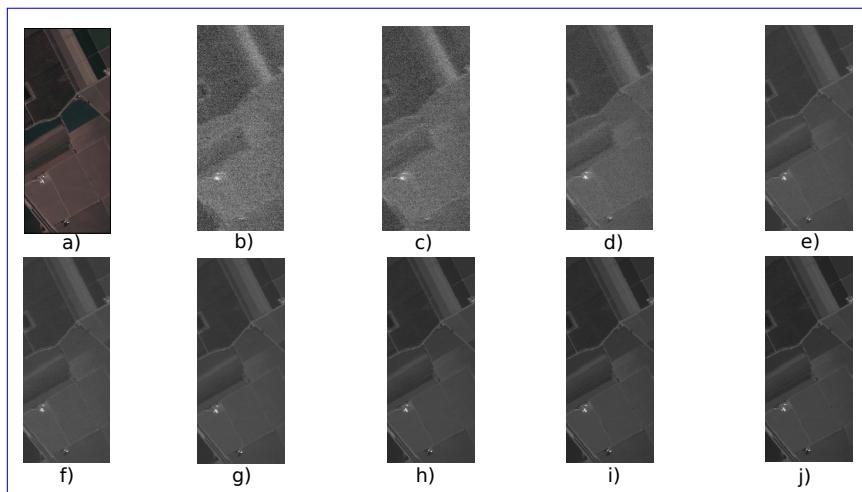


Figure 9. Original Sentinel-2 spectral bands.

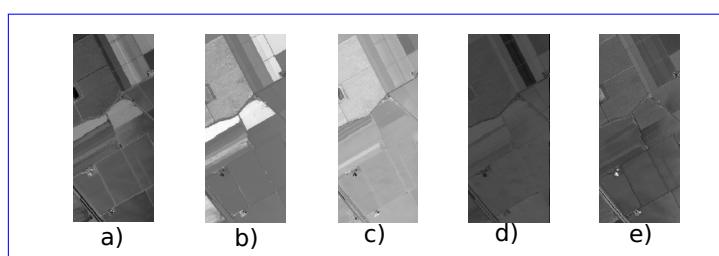


Figure 10. Tucker Decomposition Tensor bands.

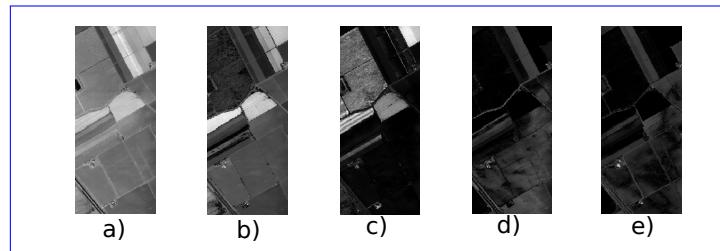


Figure 11. Nonnegative Tucker Decomposition Tensor bands.

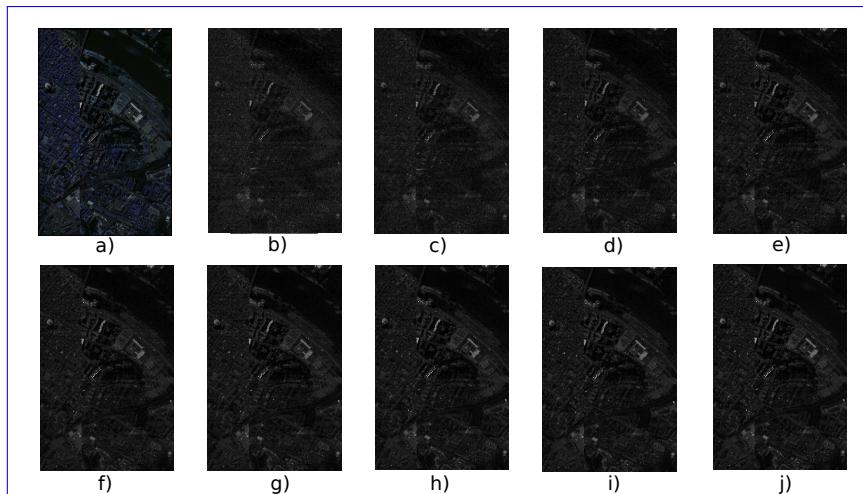


Figure 12. Original Sentinel-2 spectral bands.

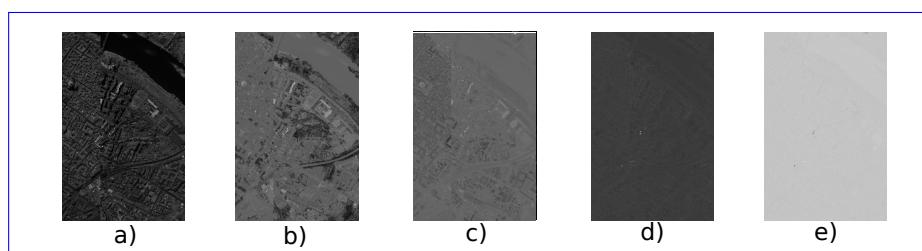


Figure 13. Tucker Decomposition Tensor bands.

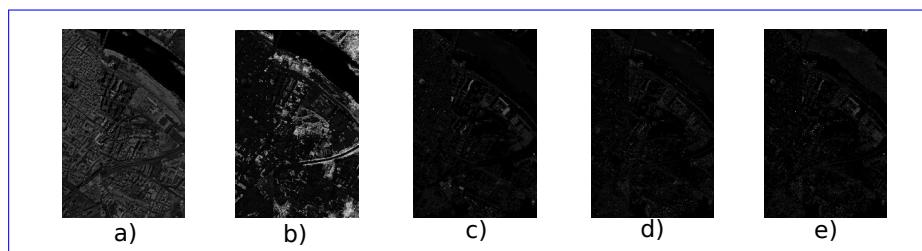


Figure 14. Nonnegative Tucker Decomposition Tensor bands.

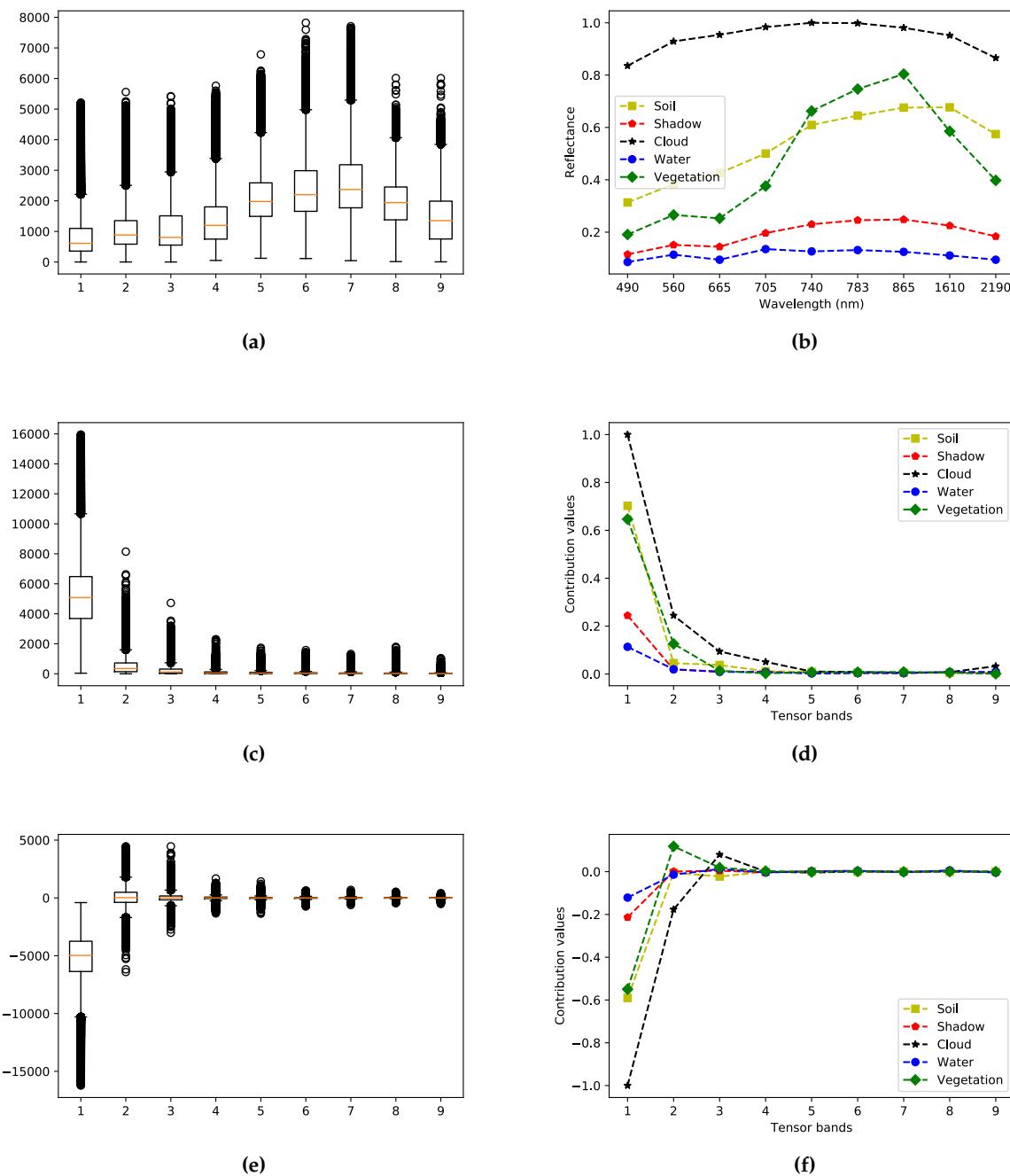


Figure 15. Box and whiskers plot and spectral signatures for a) and b) original spectral bands, c) and d) tensor bands of the NTKD and e) and f) tensor bands of the TKD

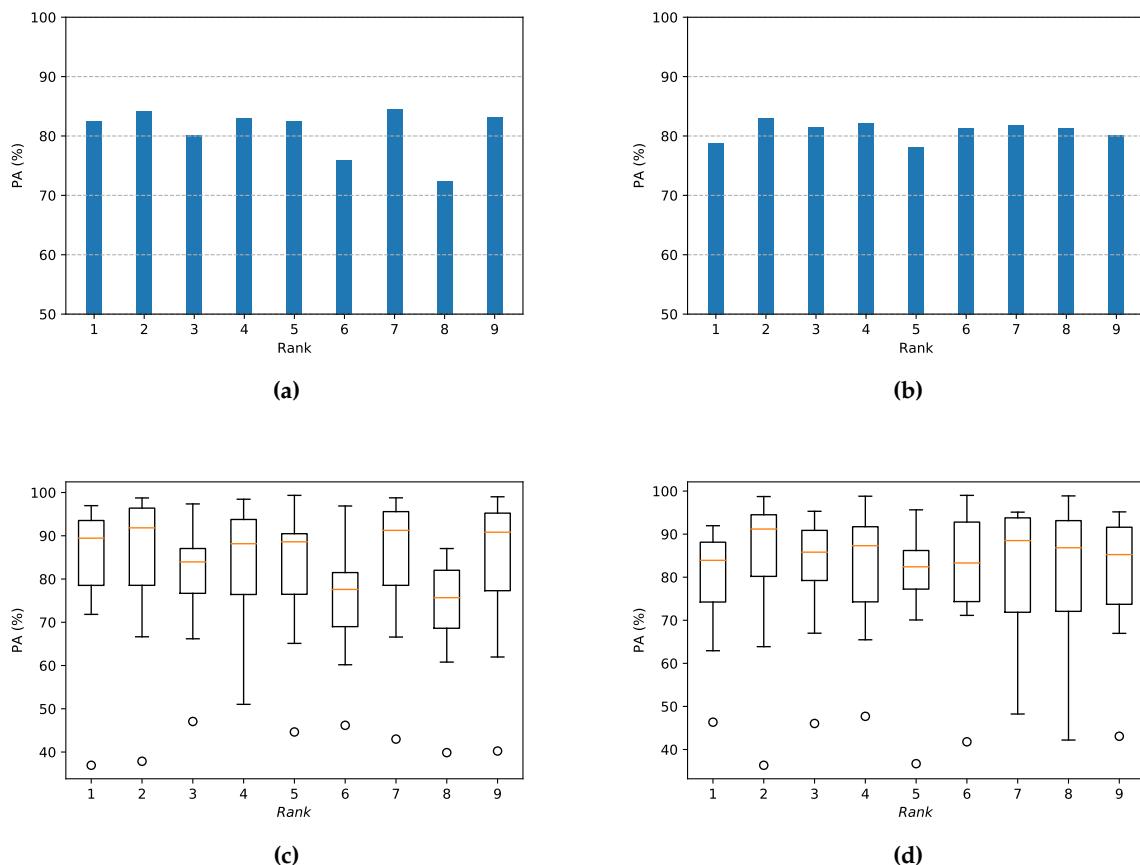


Figure 16. Pixel accuracy vs Rank results a) Comparative bar plot NTKD and TKD, b) Comparative bar plot NTKD and TKD c) Box and whiskers plot for NTKD, and d) Box and whiskers plot for TKD

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

277 Funding: This work was supported by the National Council of Science and Technology CONACYT of Mexico
278 under grant XXXXXXXX.

279 Acknowledgments:

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

281 Abbreviations

282 The following abbreviations are used in this manuscript:

283	ANN	Artificial Neural Network
	CNN	Convolutional neural network
	CPD	Canonical Polyadic Decomposition
284	DL	Deep Learning
	FCN	Fully Convolutional Network
	HOOI	Higher-Order Orthogonal Iteration
	HOSVD	Higher-Order Singular Value Decomposition

285 References

- 286** 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.

- 289 2. He, Z.; Hu, J.; Wang, Y. Low-rank tensor learning for classification of hyperspectral image with limited
290 labeled sample. *IEEE Signal Process.* **2017**, *145*, 12–25.
- 291 3. Richards, A.; Xiuping, J.J. Band selection in sentinel-2 satellite for agriculture applications. In *Remote Sensing*
292 *Digital Image Analysis*, 4th ed.; Springer-Verlag: Berlin, Germany, 2006.
- 293 4. Zhang, T.; Su, J.; Liu, C.; Chen, W.; Liu, H.; Liu, G. Band selection in sentinel-2 satellite for agriculture
294 applications. In Proceedings of the 23rd International Conference on Automation & Computing, University
295 of Huddersfield, Huddersfield, UK, 7–8 September 2017.
- 296 5. Xie, Y.; Zhao, X.; Li, L.; Wang, H. Calculating NDVI for Landsat7-ETM data after atmospheric correction
297 using 6S model: A case study in Zhangye city, China. In Proceedings of the 18th International Conference on
298 Geoinformatics, Beijing, China, 18–20 June 2010.
- 299 6. Gao, B. NDWI—A normalized difference water index for remote sensing of vegetation liquid water from
300 space. *Remote Sens. Environ.* **1996**, *58*, 1–6.
- 301 7. Ham, J.; Chen, Y.; Crawford, M.; Ghosh, J. Investigation of the random forest framework for classification of
302 hyperspectral data. *IEEE Trans. Geosci. Remote Sens.* **2005**, *43*, 492–501.
- 303 8. Hearst, Marti A. Support Vector Machines. *IEEE Intell. Syst.* **1998**, *13*, 18–28.
- 304 9. Huang, X.; Zhang, L. An SVM Ensemble Approach Combining Spectral, Structural, and Semantic Features
305 for the Classification of High-Resolution Remotely Sensed Imagery. *IEEE Trans. Geosci. Remote Sens.*
306 **2013**, *51*, 257–272.
- 307 10. Delalieux, S.; Somers, B.; Haest, B.; Spanhove, T.; Vanden Borre, J.; Mucher, S. Heathland conservation
308 status mapping through integration of hyperspectral mixture analysis and decision tree classifiers.
Remote Sens. Environ. **2012**, *126*, 222–231.
- 309 11. Kemker, R.; Salvaggio, C.; Kanan, C. Algorithms for semantic segmentation of multispectral remote sensing
310 imagery using deep learning. *ISPRS J. Photogramm. Remote Sens.* **2018**, *145*, 60–77.
- 311 12. Pirotti, F.; Sunar, F.; Piragnolo, M. Benchmark of machine learning methods for classification of a sentinel-2
312 image. In Proceedings of the XXIII ISPRS Congress, Prague, Czech Republic, 12–19 July 2016.
- 313 13. Mateo-García, G.; Gómez-Chova, L.; Camps-Valls, G. Convolutional neural networks for multispectral image
314 cloud masking. In Proceedings of the IGARSS, Fort Worth, TX, USA, 23–28 July 2017.
- 315 14. Guo, X.; Huang, X.; Zhang, L.; Zhang, L.; Plaza, A.; Benediktsson, J. A. Support Tensor Machines for
316 Classification of Hyperspectral Remote Sensing Imagery. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 3248–3264.
- 317 15. Cichocki, A.; Mandic, D.; De Lathauwer, L.; Zhou, G.; Zhao, Q.; Caiafa, C.; Phan, H. Tensor Decompositions
318 for Signal Processing Applications: From two-way to multiway component analysis. *IEEE Signal Process. Mag.*
319 **2015**, *32*, 145–163.
- 320 16. Jolliffe, I.T. *Principal Component Analysis*, 2nd ed.; Springer Verlag: New York, NY, USA, 2002.
- 321 17. Kolda, T.; Bader, B. Tensor Decompositions and Applications. *SIAM Rev.* **2009**, *51*, 455–500.
- 322 18. Lopez, J.; Santos, S.; Torres, D.; Atzberger, C. Convolutional Neural Networks for Semantic Segmentation
323 of Multispectral Remote Sensing Images. In Proceedings of the LATINCOM, Guadalajara, Mexico,
324 14–16 November 2018.
- 325 19. European Space Agency. Available online: <https://sentinel.esa.int/web/sentinel/missions/sentinel-2>
326 (accessed on 15 July 2019).
- 327 20. Kemker, R.; Kanan, C. Deep Neural Networks for Semantic Segmentation of Multispectral Remote Sensing
328 Imagery. *arXiv* **2017**, arXiv:abs/1703.06452.
- 329 21. Hamida, A.; Benoît, A.; Lambert, P.; Klein, L.; Amar, C.; Audebert, N.; Lefèvre, S. Deep learning for
330 semantic segmentation of remote sensing images with rich spectral content. In Proceedings of the IGARSS,
331 Fort Worth, TX, USA, 23–28 July 2017.
- 332 22. Wang, Q.; Lin, J.; Yuan, Y. Salient Band Selection for Hyperspectral Image Classification via Manifold
333 Ranking. *IEEE Trans. Neural Netw. Learn. Syst.* **2016**, *27*, 1279–1289.
- 334 23. Li, S.; Qiu, J.; Yang, X.; Liu, H.; Wan, D.; Zhu, Y. A novel approach to hyperspectral band selection based on
335 spectral shape similarity analysis and fast branch and bound search. *Eng. Appl. Artif. Intell.* **2014**, *27*, 241–250.
- 336 24. Zhang, L.; Zhang, L.; Tao, D.; Huang, X.; Du, B. Compression of hyperspectral remote sensing images by
337 tensor approach. *Neurocomputing* **2015**, *147*, 358–363.
- 338 25. Astrid, M.; Lee, Seung-Ik. CP-decomposition with Tensor Power Method for Convolutional Neural Networks
339 compression. In Proceedings of the BigComp, Jeju, Korea, 13–16 February 2017.
- 340 26. Chien, J.; Bao, Y. Tensor-factorized neural networks. *IEEE Trans. Neural Networks Learn. Syst.* **2018**, *29*, 1998–2011.

- 342 27. An, J.; Lei, J.; Song, Y.; Zhang, X.; Guo J. Tensor Based Multiscale Low Rank Decomposition for Hyperspectral
343 Images Dimensionality Reductio. *Remote Sens.* **2019**, *11*, 1485.
- 344 28. Li, J.; Liu, Z. Multispectral Transforms Using Convolution Neural Networks for Remote Sensing Multispectral
345 Image Compression. *Remote Sens.* **2019**, *11*, 759.
- 346 29. An, J.; Song, Y.; Guo, Y.; Ma, X.; Zhang, X. Tensor Discriminant Analysis via Compact Feature Representation
347 for Hyperspectral Images Dimensionality Reduction. *Remote Sens.* **2019**, *11*, 1822.
- 348 30. Absil, P.-A.; Mahony, R.; Sepulchre, R. *Optimization Algorithms on Matrix Manifolds*, 1st ed.; Princeton
349 University Press: Princeton, NJ, USA, 2007.
- 350 31. De Lathauwer, L.; De Moor, B.; Vandewalle, J. On the best rank-1 and rank-(R_1, R_2, \dots, R_N) approximation
351 of higher-order tensors. *SIAM J. Matrix Anal. Appl.* **2000**, *21*, 1324–1342.
- 352 32. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*, 1st ed.; MIT Press, 2016.
- 353 33. Sheehan, B. N.; Saad, Y. Higher Order Orthogonal Iteration of Tensors (HOOI) and its Relation to PCA and
354 GLRAM. In Proceedings of the 7th SIAM International Conference on Data Mining, Minneapolis, MN, USA,
355 26–28 April 2007.
- 356 34. Badrinarayanan, V.; Kendall, A.; Cipolla, R. SegNet: A Deep Convolutional Encoder-Decoder Architecture
357 for Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 2481–2495.
- 358 35. De Lathauwer, L.; De Moor, B.; Vandewalle, J. A Multilinear Singular Value Decomposition. *SIAM J. Matrix
359 Anal. Appl.* **2000**, *21*, 1253–1278.
- 360 36. Rodes, I.; Inglada, J.; Hagolle, O.; Dejoux, J.; Dedieu, G. Sampling strategies for unsupervised classification
361 of multitemporal high resolution optical images over very large areas. In Proceedings of the 2012 IEEE
362 International Geoscience and Remote Sensing Symposium, Munich, Germany, 22–27 July 2012.

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

364 © 2020 by the authors. Submitted to *Remote Sens.* for possible open access publication
365 under the terms and conditions of the Creative Commons Attribution (CC BY) license
366 (<http://creativecommons.org/licenses/by/4.0/>).