Tensor-Based Classification Models for Hyperspectral Data Analysis

Konstantinos Makantasis, Anastasios D. Doulamis, Nikolaos D. Doulamis, and Antonis Nikitakis

***Abstract*— In this paper, we present tensor-based linear and nonlinear models for hyperspectral data classification and analysis. By exploiting the principles of tensor algebra, we introduce new classification architectures, the weight parameters of which satisfy the rank-1 canonical decomposition property. Then, we propose learning algorithms to train both linear and nonlinear classifiers. The advantages of the proposed classification approach are that: 1) it significantly reduces the number of weight parameters required to train the model (and thus the respective number of training samples); 2) it provides a physical interpretation of model coefficients on the classification output; and 3) it retains the spatial and spectral coherency of the input samples. The linear tensor-based model exploits the principles of logistic regression, assuming the rank-1 canonical decomposition property among its weights. For the nonlinear classifier, we propose a modification of a feedforward neural network (FNN), called rank-1 FNN, since its weights satisfy again the rank-1 canonical decomposition property. An appropriate learning algorithm is also proposed to train the network. Experimental results and comparisons with state-of-the-art classification methods, either linear (e.g., linear support vector machine) or nonlinear (e.g., deep learning), indicate the outperformance of the proposed scheme, especially in the cases where a small number of training samples is available.**

***Index Terms*— Dimensionality reduction, hyperspectral data analysis, nonlinear modeling, rank-1 feedforward neural networks (FNNs), tensor-based classification.**

I. Introduction

THE recent advances in optics and photonics have stimulated the deployment of hyperspectral imaging sensors of high-spatial and high-spectral resolution. These sensors are now placed on satellite, unmanned aerial vehicle, and ground acquisition platforms used for material, object, and terrain land detection and classification [1]. Although high-spatial and high-spectral resolution improves classification accuracy, it also imposes several research challenges derived as a consequence of the so-called "curse of dimensionality"; the difficulties arise when we need to analyze and organize data in high-dimensional spaces. Hyperspectral data have their own unique characteristics, though being applied for a wide variety of applications, such as agriculture, surveillance, astronomy, and biomedical imaging [2]: 1) high-dimensional data; 2) limited number of labeled samples; and 3) large spatial variability of spectral signatures [3].

Most of existing works, concerning hyperspectral image classification, follow the conventional workflow of pattern recognition process, consisting of two separate steps. First, features are extracted from the raw data, creating labeled training data sets. Second, classifiers, linear or nonlinear, such as support vector machines (SVMs) and neural networks (NNs) [4], are used to map the extracted features to the target (desired) outputs. The key problem, however, in applying such conventional processes in classifying high-dimensional hyperspectral data is that a large number of labeled training samples are required to model the statistical input diversities and consequently to well train the classifier. In remote sensing applications, a collection of a large number of labeled data is an expensive and time-consuming process. Another drawback is that classifiers are often used as "black boxes" [5]. This means that there is no direct interpretation of how spatial and spectral bands contribute to the final classification outcome.

One way to address issues deriving from the high dimensionality and heterogeneity of the data is to employ statistical learning methods [6]. However, even in this case, the problem of extracting a set of appropriate features remains. Feature representation significantly affects the classification outcome. Estimating a suitable set of discriminative features, in order to increase the accuracy of the classifier, is an arduous task especially when the data is lying in high-dimensional spaces.

For this reason, recently deep learning paradigms have been investigated for classifying hyperspectral data [7]–[10]. Deep learning machines receive as inputs, instead of features, the raw sensory data. Then, they nonlinearly transform the raw inputs to hierarchies of representations which are used, in the following, as "the most suitable features" in a supervised mapping phase. Thus, deep learning tackles feature-related issues. This is also proven by the current research outcomes [11]–[14] indicating the outperformance of deep learning machines in accurately detecting various objects in hyperspectral imaging data. Examples include the detection of man-made constructions rather than natural ones [15], [16], vehicles' detection [17], object tracking [18], land cover mapping [19], and critical infrastructure assessment [20].

However, a typical deep learning architecture contains a huge number of tunable parameters implying that a large number of samples is also needed to accurately train the network.

Manuscript received October 6, 2017; revised March 27, 2018; accepted May 23, 2018. Date of publication July 9, 2018; date of current version November 22, 2018. This work was supported by the European Union through the H2020 STOP-IT Project, Strategic, Tactical, Operational Protection of Water Infrastructure Against Cyber-Physical Threats, under Grant 740610. *(Corresponding author: Konstantinos Makantasis.)*

K. Makantasis is with the KIOS Research and Innovation Center of Excellence, 1678 Nicosia, Cyprus (e-mail: konst.makantasis@gmail.com).

A. D. Doulamis and N. D. Doulamis are with the National Technical University of Athens, 15780 Athens, Greece and also with the Institute of Communications and Computer Systems, Athens, Greece.

A. Nikitakis is with Althexis Solutions Ltd., 1066 Nicosia, Cyprus.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TGRS.2018.2845450

In addition, deep learning processes present high computational complexity.

Tensor-based machine learning models are promising alternatives for hyperspectral data classification [21], [22]. In particular, Zhou *et al.* [21] and Tan *et al.* [22] have introduced a linear model using tensor-based regression with applications in neuroimaging data analysis [21] and for classification [22]. These approaches are considered as the first works which discussed the statistical inference procedure for the general raw tensor regression. In conventional learning models, usually the inputs are vector data. Therefore, in the case of multidimensional input arrays, first tensor vectorization is carried out. However, vectorization destroys the inherent spatial and spectral structure of the input which can offer a physical interpretation of how spatial information and spectral bands contribute to the classification outcome. Furthermore, tensor vectorization fails to address the issues that stem from the high dimensionality of the data, since again a large number of tunable parameters are required. To handle these limitations, we need to consider the input data as tensors, keeping the spatial and spectral structure of the data, and then, using principles of tensor algebra, to find out ways to reduce the number of parameters needed to be estimated during training.

A. Our Contribution

In this paper, we propose a new machine learning model which receives as inputs the multidimensional tensor signals as they are the hyperspectral images, i.e., 3-D tensor cubes. The model weight parameters satisfy the rank-1 canonical decomposition property meaning that the weight parameters are decomposed as a linear combination of a minimal number of possibly nonorthogonal rank-1 terms [23]. Using such decomposition of the model weights, we are able to significantly reduce the number of parameters required to train the classifier. Thus, a smaller labeled data set is needed than in conventional learning approaches where the tensor inputs are first vectorized.

Another advantage of the rank-1 canonical decomposition property for the model weights is that it retains the structure of the spatial and spectral band information, which is a very important aspect for hyperspectral data classification. This is due to the fact that it actually permits the extraction of valuable information regarding the contribution of each of the hyperspectral bands to the classification. Thus, the proposed canonical decomposition provides a physical interpretation of the classification outcome, i.e., how the location of the pixels (spatial information) and the spectral bands (spectral information) influence the final classification performance.

This paper is motivated from [21] which proposes a single linear output tensor regression model for binary classification. In contrast to [21], in this paper, a multiclass classification problem is investigated using tensor-based logistic regression models of multiple outputs. In addition, the rank-1 canonical decomposition property is also applied, apart from high-order linear to nonlinear classifiers, which is not a straightforward process. The proposed high-order nonlinear model relies on a modification of a feedforward NN (FNN), while it retains the universal approximation principles; capability of the network to approximate any unknown function, under some assumptions of continuity, within any degree of accuracy. The main difference is that the model weights satisfy the rank-1 canonical decomposition property. Therefore, the number of parameters (and consequently the number of training samples) is significantly reduced, especially for the cases where tensor inputs are considered. A new learning algorithm is also introduced to train the network without spoiling the canonical decomposition assumption. We call the proposed high-order nonlinear model as rank-1-FNN.

To sum up, the main contribution of this paper is threefold. First, we introduce new learning models, linear and nonlinear, that consider a rank-1 canonical decomposition of their weight parameters. This way, a quite smaller number of weights are required compared with conventional learning paradigms. This, in the sequel, requires a smaller number of samples used to train the model, which is in line with remote sensing applications, where a limited number of samples are available. The new models are suitable for tensor input data of high dimensions. Second, the rank-1 canonical decomposition property allows for a physical interpretation of how each spatial and spectral band of the tensor inputs affects the classification outcome. This is an important attribute in analyzing hyperspectral data. Third, the introduction of a rank-1 FNN nonlinear classifier allows for modeling of complex relationships, due to the universal approximation property of neural nets [24] while simultaneously keeping the aforementioned advantages.

The rest of this paper is organized as follows. Section II presents the problem formulation, as well as, the notation and tensor algebra operations that will be used throughout this paper. Sections III and IV present the development of the high-order linear and nonlinear models. Experimental evaluation of the developed models is presented in Section V, and Section VI concludes this paper.

II. Problem Formulation and Tensor Algebra Notation

A. Problem Formulation

Let us denote as $\boldsymbol{X}_i$ the ith patch of a hyperspectral image that we would like to classify into one of C available classes. Each class expresses, for example, the type of vegetation or soil properties for the patch $\boldsymbol{X}_i$. In this paper, we consider $\boldsymbol{X}_i$ as a 3-D tensor, i.e., $\boldsymbol{X}_i \in \mathbb{R}^{p_1 \times p_2 \times p_3}$, where variables p_1 and p_2 refer to the spatial dimensions of the hyperspectral patch and p_3 to the number of spectral bands. In a more general case, variable $\boldsymbol{X}_i$ can be seen as a D-dimensional tensor, that is, $\boldsymbol{X}_i \in \mathbb{R}^{p_1 \times \cdots \times p_D}$.

Let us also denote as $p_w^k(\boldsymbol{X}_i)$, with $k = 1, \ldots, C$ a relationship (linear or nonlinear) that expresses the probability of the observation $\boldsymbol{X}_i$ to belong to the kth class. Subscript w indicates the dependence of the relationship $p_w^k(\cdot)$ on weight parameters. Aggregating the values $p_w^k(\cdot)$ over all C classes, we form a classification vector, say $\boldsymbol{y}_i$, the elements of which $y_{i,k} \equiv p_w^k(\cdot)$. Then, the maximum probability value over all k classes indicates the class to which the hyperspectral image

patch belongs to

$$\hat{k} = \arg\max_{\forall k}\{y_{i,k} \equiv p_w^k(\boldsymbol{X}_i)\}. \quad (1)$$

The values of $y_{i,k}$ are estimated using machine learning algorithms. For this reason, a training data set consisting N samples is considered

$$\mathcal{S} = \{(\boldsymbol{X}_i, \boldsymbol{t}_i)\}_{i=1}^N \quad (2)$$

where $\boldsymbol{t}_i$ is a C-dimensional vector, the elements of which $t_{i,j}$ are all zero except for one which equals unity indicating the class to which $\boldsymbol{X}_i$ belongs to. That is, $\boldsymbol{t}_i \in \{0,1\}^C$ and $\sum_{j=1}^C t_{i,j} = 1$. In the following, we omit subscript i for simplicity purposes if we refer to an input sample.

Multidimensional arrays are also known as tensors. Since tensors are a key concept of the proposed high-order learning model (linear and nonlinear), in the following, some basic notations and definitions regarding tensor algebra are presented that will be used throughout this paper.

B. Tensor Algebra Notations and Definitions

In this paper, tensors are denoted with bold uppercase letters, vectors with bold lowercase letters, and scalars with lowercase letters.

1) Tensor Vectorization: The vec($\boldsymbol{B}$) operator stacks the entries of a D-dimensional tensor $\boldsymbol{B} \in \mathbb{R}^{p_1 \times \cdots \times p_D}$ on a column vector. Specifically, an entry $\boldsymbol{B} = [\cdots b_{i_1,\cdots,i_D} \cdots]$ maps to the jth entry of vec($\boldsymbol{B}$), in which $j = 1 + \sum_{d=1}^D (i_d - 1) \prod_{d'=1}^{d-1} p_{d'}$.

2) Tensor Products: Given two tensors $\boldsymbol{A} = [\boldsymbol{a}_1 \cdots \boldsymbol{a}_n] \in \mathbb{R}^{m \times n}$ and $\boldsymbol{B} = [\boldsymbol{b}_1 \cdots \boldsymbol{b}_q] \in \mathbb{R}^{p \times q}$, the following products can be defined.

1) *Kronecker Product:* The Kronecker product is the $mp \times nq$ matrix

$$\boldsymbol{A} \otimes \boldsymbol{B} = [\boldsymbol{a}_1 \otimes \boldsymbol{B} \cdots \boldsymbol{a}_n \otimes \boldsymbol{B}] = \begin{bmatrix} a_{11}\boldsymbol{B} & \cdots & a_{1n}\boldsymbol{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\boldsymbol{B} & \cdots & a_{mn}\boldsymbol{B} \end{bmatrix}.$$

2) *Khatri–Rao Product:* The Khatri–Rao product $\boldsymbol{A} \odot \boldsymbol{B}$ is defined as the $mp \times n$ columnwise Kronecker product, i.e., $\boldsymbol{A} \odot \boldsymbol{B} = [\boldsymbol{a}_1 \otimes \boldsymbol{b}_1 \;\; \boldsymbol{a}_2 \otimes \boldsymbol{b}_2 \cdots \boldsymbol{a}_n \otimes \boldsymbol{b}_n]$, if $\boldsymbol{A}$ and $\boldsymbol{B}$ have the same number of columns, i.e., $n = q$.

3) *Outer Product:* The outer product $\boldsymbol{b}^{(1)} \circ \boldsymbol{b}^{(2)} \circ \cdots \circ \boldsymbol{b}^{(N)}$ of the vectors $\{\boldsymbol{b}^{(i)}\}_{i=1}^N$ forms a tensor whose element at $(i_1, i_2, \ldots, i_N)$ position equals $\prod_{j=1}^N b_{i_j}^{(j)}$.

3) Tensor Matricization: The mode-d matricization, $\boldsymbol{B}_{(d)}$, maps a tensor $\boldsymbol{B}$ to a $p_d \times \prod_{d' \neq d} p_{d'}$ matrix by arranging the mode-d fibers to be the columns of the resulting matrix. In particular, the $(i_1, \ldots, i_D)$ element of $\boldsymbol{B}$ maps to the (i_d, j) element of $\boldsymbol{B}_{(d)}$, where $j = 1 + \sum_{d' \neq d} (i_{d'} - 1) \prod_{d'' < d', d'' \neq d} p_{d''}$.

4) Rank-R Decomposition: A tensor $\boldsymbol{B} \in \mathbb{R}^{p_1 \times \cdots \times p_D}$ admits a rank-R decomposition if $\boldsymbol{B} = \sum_{r=1}^R \boldsymbol{b}_1^{(r)} \circ \cdots \circ \boldsymbol{b}_D^{(r)}$, where the symbol $''\circ''$ represents the vector outer product and $\boldsymbol{b}_d^{(r)} \in \mathbb{R}^{p_d}$, $d = 1, \ldots, D$, $r = 1, \ldots, R$. The decomposition can be represented by $\boldsymbol{B} = [\![\boldsymbol{B}_1, \ldots, \boldsymbol{B}_D]\!]$, where $\boldsymbol{B}_d = [\boldsymbol{b}_d^{(1)}, \ldots, \boldsymbol{b}_d^{(R)}] \in \mathbb{R}^{p_d \times R}$, $d = 1, \ldots, D$. When a tensor $\boldsymbol{B}$ admits a rank-R decomposition, the following results hold:

$$\boldsymbol{B}_{(d)} = \boldsymbol{B}_d (\boldsymbol{B}_D \odot \cdots \odot \boldsymbol{B}_{d+1} \odot \boldsymbol{B}_{d-1} \odot \cdots \odot \boldsymbol{B}_1)^T \quad (3)$$

and

$$\text{vec}(\boldsymbol{B}) = (\boldsymbol{B}_D \odot \cdots \odot \boldsymbol{B}_1)\boldsymbol{1}_R \quad (4)$$

where $\boldsymbol{1}_R$ is a vector of R ones. For more information regarding tensor algebra, please refer to [25].

III. High-Order Linear Modeling

A. Vector Logistic Regression

Let us first assume for simplicity that the input samples are of vector forms. We denote these input vectors as $\boldsymbol{x} \in \mathbb{R}^{p_1}$, where variable p_1 expresses vector dimension. Using the logistic regression framework [26], one can model the probability function $p_w^k(\cdot)$ as

$$p_w^k(\boldsymbol{x}) = \frac{\exp \boldsymbol{w}^{(k)T}\boldsymbol{x}}{\sum_{i=1}^C \exp \boldsymbol{w}^{(i)T}\boldsymbol{x}} \quad (5)$$

where $\boldsymbol{w}^{(k)} \in \mathbb{R}^{p_1}$ with $k = 1, 2, \ldots, C$ stands for the weights with respect to the kth class. Matrix $\boldsymbol{W} = [\boldsymbol{w}^{(1)}\boldsymbol{w}^{(2)} \cdots \boldsymbol{w}^{(C)}]$ includes all the weights involved in the model. The rational meaning of the weight parameters $\boldsymbol{w}^{(k)}$ is that they express the degree of confidence of the vector input $\boldsymbol{x}$ to belong to the kth class out of C available classes. In addition, the elements $w_j^{(k)}$ of $\boldsymbol{w}^{(k)} = [\cdots w_j^{(k)} \cdots]^T$ express the degree of significance of each element of the input vector $\boldsymbol{x}$ with respect to the kth class.

One simple way to extend (5) in the case that the inputs are tensors, i.e., $\boldsymbol{X}_i \in \mathbb{R}^{p_1 \times \cdots \times p_D}$, is to take the vectorized forms of them (see the respective text on algebra notation of Section II-B1). The main limitation of such an approach is that: 1) a large number of parameters is needed to be estimated, particularly ($C \prod_{l=1}^D p_l$) and 2) vectorization spoils the spatial structure of tensor inputs, that is, pixels belonging to a neighboring region frequently present similar properties (spatial coherency). A large number of parameters also imply a large number of available labeled observations in order to successfully complete the training procedure. However, usually the available labeled samples are limited due to manual effort required to collect and annotate them.

B. Matrix Logistic Regression

In the case of matrix input observations, $\boldsymbol{X} \in \mathbb{R}^{p_1 \times p_2}$, one can reduce the number of model parameters by taking into consideration the concepts of [27], applied for electroencephalogram data classification. Then, the logistic regression model is given by

$$p_w^k(\boldsymbol{X}) = \frac{\exp \boldsymbol{w}_1^{(k)T}\boldsymbol{X}\boldsymbol{w}_2^{(k)}}{\sum_{i=1}^C \exp \boldsymbol{w}_1^{(i)T}\boldsymbol{X}\boldsymbol{w}_2^{(i)}} \quad (6)$$

where $\boldsymbol{w}_1^{(i)} \in \mathbb{R}^{p_1}$ and $\boldsymbol{w}_2^{(i)} \in \mathbb{R}^{p_2}$. We also define as $\boldsymbol{W}_1 = [\boldsymbol{w}_1^{(1)}\boldsymbol{w}_1^{(2)} \cdots \boldsymbol{w}_1^{(C)}]$ and $\boldsymbol{W}_2 = [\boldsymbol{w}_2^{(1)}\boldsymbol{w}_2^{(2)} \cdots \boldsymbol{w}_2^{(C)}]$ the aggregate model parameters.

The key advantage of (6) is that the number of required parameters is $C(p_1 + p_2)$ instead of $C(p_1 \cdot p_2)$ that would be needed if one vectorizes matrix observation input data $\boldsymbol{X}$. This means that the weight parameters $\boldsymbol{w}^{(k)}$, in relation (6),

for the kth class are rank-1 canonically decomposed into the weights of $\boldsymbol{w}_1^{(k)} \in \mathbb{R}^{p_1}$ and $\boldsymbol{w}_2^{(k)} \in \mathbb{R}^{p_2}$, i.e.,

$$\boldsymbol{w}^{(k)} = \boldsymbol{w}_2^{(k)} \otimes \boldsymbol{w}_1^{(k)} \tag{7}$$

where operator $\otimes$ is the Kronecker product as defined in 1) in Section II-B2.

Using (7), one can write (6) as

$$p_w^k(\boldsymbol{X}) = \frac{\exp[(\boldsymbol{w}_2^{(k)} \otimes \boldsymbol{w}_1^{(k)})^T \mathrm{vec}(\boldsymbol{X})]}{\sum_{i=1}^{C} \exp[(\boldsymbol{w}_2^{(i)} \otimes \boldsymbol{w}_1^{(i)})^T \mathrm{vec}(\boldsymbol{X})]}. \tag{8}$$

C. Tensor-Based Logistic Regression

The aforementioned concept can be extended in the case of tensor inputs, $\boldsymbol{X} \in \mathbb{R}^{p_1 \times \cdots \times p_D}$ [21]. In this way, we have that

$$p_w^k(\boldsymbol{X}) = \frac{\exp\left[\left(\boldsymbol{w}_D^{(k)} \otimes \cdots \otimes \boldsymbol{w}_1^{(k)}\right)^T \mathrm{vec}(\boldsymbol{X})\right]}{\sum_{i=1}^{C} \exp\left[\left(\boldsymbol{w}_D^{(i)} \otimes \cdots \otimes \boldsymbol{w}_1^{(i)}\right)^T \mathrm{vec}(\boldsymbol{X})\right]} \tag{9}$$

where $\boldsymbol{w}_l^{(k)} \in \mathbb{R}^{p_l}$ with $l = 1, 2, \ldots, D$ is the lth rank-1 canonical decomposition of the weight parameters $\boldsymbol{w}^{(k)}$ for the kth class. This property is referred as CANDECOMP, stem from CANonical DECOMPosition, or PARAFAC, stem from PARAllel FACtors, in the literature [28], [29]. That is

$$\boldsymbol{w}^{(k)} = \boldsymbol{w}_D^{(k)} \otimes \cdots \otimes \boldsymbol{w}_1^{(k)}. \tag{10}$$

Equation (9) can be seen as an expression of the Khatri–Rao product, denoted as operator $\odot$ (see Section II), which is the columnwise Kronecker product of the rank-1 canonical decomposition weight parameters $\boldsymbol{w}_l^{(k)}$

$$p_w^k(\boldsymbol{X}) = \frac{\exp[(\boldsymbol{w}_D^{(k)} \odot \cdots \odot \boldsymbol{w}_1^{(k)})^T \mathrm{vec}(\boldsymbol{X})]}{\sum_{i=1}^{C} \exp[(\boldsymbol{w}_D^{(i)} \odot \cdots \odot \boldsymbol{w}_1^{(i)})^T \mathrm{vec}(\boldsymbol{X})]}. \tag{11}$$

In (9) and (11), we can aggregate the total weight parameters as

$$\boldsymbol{W}_l = \left[\boldsymbol{w}_l^{(1)} \boldsymbol{w}_l^{(2)} \cdots \boldsymbol{w}_l^{(C)}\right] \tag{12}$$

with $l = 1, 2, \ldots, D$. In other words, matrix $\boldsymbol{W}_l$ contains all the weight parameters with respect to the lth dimension of the tensor $\boldsymbol{X}$ over all classes, i.e., related with the p_l tensor dimension.

The representation of (9) and (11) significantly reduces the number of model parameters needed for classifying the tensor inputs $\boldsymbol{X}$. In particular, the vector-based logistic regression model derived through vectorization of tensor $\boldsymbol{X}$ requires the estimation of $C \prod_{l=1}^{D} p_l$ parameters, whereas the tensor-based representation of (9) and (11) reduces this number to $C \sum_{l=1}^{D} p_l$.

The advantages of the aforementioned proposed representation for the classification of hyperspectral images are twofold. First, as we reduce the number of weight parameters, a smaller number of labeled data samples are required to train the logistic regression model. This is an important factor for developing classifiers that can better generalize to unseen hyperspectral data. Usually, the manual effort for the annotation is laborious and therefore a small number of labeled training data are available. Second, the rank-1 canonical decomposition of the model weights provides a physical interpretation of how the spatial and spectral information of the hyperspectral tensor input $\boldsymbol{X}$ contributes to the classification. In particular, according to the statements made right after (5), the weights $\boldsymbol{w}^{(k)}$ express the degree of importance of the tensor input $\boldsymbol{X}$ to belong to the kth class. Since these weights are canonically decomposed into D separate weights $\boldsymbol{w}_l^{(k)}$, each indicating the contribution of p_l tensor dimension that belongs to the kth class, the proposed model provides a quantitative representation of how the elements of each tensor dimension tunes the classification performance.

More specifically, in the case of hyperspectral imaging, the dimension of tensor inputs equals 3, i.e., $D = 3$. The first two dimensions refer to the spatial properties of the pixel data, whereas the third one to the spectral bands. Therefore, the first two decomposed weights, i.e., $\boldsymbol{w}^{(1)}$ and $\boldsymbol{w}^{(2)}$, express how the pixel spatial coherency affects the classification outcome. On the other hand, the third decomposed weight vector $\boldsymbol{w}^{(3)}$ indicates how the spectral bands influence the classification and which of the spectral bands are the most salient. This property of the proposed model is very important toward the analysis of hyperspectral data, since it facilitates the interpretation of the results and quantifies the importance of the spectral bands on the classification compressing the influence of the bands that are of less importance.

D. Estimation of the Model Weights

The model weight parameters are estimated through a training set $\mathcal{S} = \{(\boldsymbol{X}_i, \boldsymbol{t}_i)\}_{i=1}^{N}$ of N samples as (2) indicates. We recall that if $\boldsymbol{X}_i$ belongs to the kth class, then $\boldsymbol{t}_i$ is a vector with all elements zero except element k, which equals 1, i.e., $t_{i,j} = 0, \forall j \neq k$ and $t_{i,k} = 1$. By minimizing the negative log-likelihood function

$$L(\boldsymbol{W}_1, \ldots, \boldsymbol{W}_D; \mathcal{S}) = -\sum_{i=1}^{N} \sum_{k=1}^{C} t_{i,k} \log p_w^k(\boldsymbol{X}_i) \tag{13}$$

the weight parameters can be estimated as

$$\{\hat{\boldsymbol{W}}_{\boldsymbol{1}}, \hat{\boldsymbol{W}}_{\boldsymbol{2}}, \cdots, \hat{\boldsymbol{W}}_{\boldsymbol{D}}\} = \arg\min_{\forall \boldsymbol{W}_i} L(\boldsymbol{W}_1, \ldots, \boldsymbol{W}_D; \mathcal{S}). \tag{14}$$

In (14), matrices $\hat{\boldsymbol{W}}_{\boldsymbol{l}}, l = 1, \ldots, D$ refer to the optimal estimates of the $\boldsymbol{W}_l$ weight parameters, expressing the contribution of the l-dimension of the tensor input to the classification for all C available classes.

Based on the statements of Section II, presenting some basic notations on tensor algebra (*rank-R decomposition and tensor matricization*), it is held that

$$\begin{aligned}\langle \boldsymbol{w}_D^{(k)} \odot \cdots \odot \boldsymbol{w}_1^{(k)}, \boldsymbol{X} \rangle \\ = \langle \boldsymbol{w}_l^{(k)}, \boldsymbol{X}_{(l)} (\boldsymbol{w}_D^{(k)} \odot \cdots \odot \boldsymbol{w}_{l+1}^{(k)} \odot \boldsymbol{w}_{l-1}^{(k)} \odot \cdots \odot \boldsymbol{w}_1^{(k)}) \rangle.\end{aligned} \tag{15}$$

The proof of (15) is given in Appendix A. In (15), $\boldsymbol{X}_{(l)}$ denotes the mode-l matricization of tensor $\boldsymbol{X}$ obtained by keeping the l-dimension intact and concatenating the slices of the rest dimensions into a long matrix [25]. We also recall that the operator $\odot$ refers to the Khatri–Rao product, while the $\langle \cdot, \cdot \rangle$ to the inner product between two tensors.

Algorithm 1 Optimal Estimation of Model Weights Using Rank-1 Canonical Decomposition

Initialization:

1. Set Iteration Index $n \rightarrow 0$
2. Randomize all the weight parameters $\boldsymbol{W}_l(n) \in \mathbb{R}^{p_l \times C}$ for all $l = 1, \ldots, D$
3. **repeat**
   - **for** $l = 1, \ldots, D$ **do**
     - 3.1 Calculate the matrix of Eq.(17), that is, $\boldsymbol{X}_{(l)}(\boldsymbol{W}_D(n) \odot \cdots \odot \boldsymbol{W}_{l+1}(n) \odot \boldsymbol{W}_{l-1}(n) \odot \cdots \odot \boldsymbol{W}_1(n))$
     - 3.2 Update matrix $\boldsymbol{W}_l(n)$ using a regression learning algorithm through minimization of (14) such as $\boldsymbol{W}_l(n+1) \rightarrow \arg\min_{\boldsymbol{W}_l} L(\boldsymbol{W}_1(n+1), \ldots, \boldsymbol{W}_{l-1}(n+1), \boldsymbol{W}_l(n), \ldots, \boldsymbol{W}_D(n))$
   - **end**
   - Set $n \rightarrow n+1$

until *termination criteria are met*;

The left-hand side of (15) expresses the arguments of $\exp(\cdot)$ involved in the linear logistic regression model of (11). Therefore, (11) can be rewritten by taking into account the right-hand side of (15). In the sequel, generation of (15) over all available classes is obtained as

$$\langle \boldsymbol{W}_D \odot \cdots \odot \boldsymbol{W}_1, \boldsymbol{X} \rangle = \operatorname{diag}(\langle \boldsymbol{W}_l, \boldsymbol{X}_{(l)}(\boldsymbol{W}_D \odot \cdots \odot \boldsymbol{W}_{l+1} \odot \boldsymbol{W}_{l-1} \odot \cdots \odot \boldsymbol{W}_1) \rangle). \tag{16}$$

As we can see, (15) is actually the inner product of two vectors. The first is the weight parameters $\boldsymbol{w}_l^{(k)}$ expressing the contribution of the l dimension of the tensor input as far as the kth class is concerned. On the other hand, the second vector is independent of the values of $\boldsymbol{w}_l^{(k)}$. Therefore, one approach for optimally estimating the weights $\boldsymbol{w}_l^{(k)}$ and consequently $\boldsymbol{w}^{(k)}$ [see (10)] is one to consider all the weight parameters $\boldsymbol{w}_q^{(k)}$ with $q \neq l$ apart from the lth fixed and then solving with respect to $\boldsymbol{w}_l^{(k)}$. This procedure is iteratively applied for all weight parameters $\boldsymbol{w}_l^{(k)}$ until some termination criteria are met. Similar statements can be concluded for the matrix representation of (16).

Actually, the learning algorithm used to optimally estimate the model weights in case that (10) is held, i.e., the rank-1 canonical decomposition simulates a regression learning where we use as inputs the data of the right-hand side of (16), i.e.,

$$\boldsymbol{X}_{(l)}(\boldsymbol{W}_D \odot \cdots \odot \boldsymbol{W}_{l+1} \odot \boldsymbol{W}_{l-1} \odot \cdots \odot \boldsymbol{W}_1). \tag{17}$$

Therefore, the parameter estimation problem can be solved by using conventional software such as scikit-learn in python.[1] The proposed learning procedure, considering the rank-1 canonical decomposition of the weight parameters, is described in Algorithm 1.

Although the high-order linear model can provide physically interpretable results, due to its structure, it is restricted to produce decision boundaries that are linear in the input space. This implies that this model is not able to cope with more complicated problems, where nonlinear decision boundaries are necessary to provide classification results of high accuracy. This motivates us to extend the previous linear regression model to a nonlinear one. The proposed nonlinear model should assume again a rank-1 canonical decomposition of the weight parameters in order to retain the aforementioned advantages in hyperspectral classification.

[1] http://scikit-learn.org/

IV. High-Order Nonlinear Modeling

The proposed high-order nonlinear model is based on the concepts of the previously discussed linear logistic regression filters with the difference that a nonlinear transformation is applied on the input data. This means, in other words, that the probability $p_w^k(\cdot)$ of an input observation $\boldsymbol{X}$ to belong to one of the C available classes is nonlinearly interwoven with respect to the input tensor data and the weight parameters that influence the importance of these data on classification performance through a function $f_w(\cdot)$, i.e.,

$$p_w^k(\cdot) = f_w(\boldsymbol{X}). \tag{18}$$

The main difficulty in implementing (18) is that function $f_w(\cdot)$ is actually unknown. One way to parameterize the unknown function $f_w(\cdot)$ is to exploit the principles of the universal approximation theorem, stating that a function, under some assumptions of continuity, can be approximated by an FNN with a finite number of neurons within any degree of accuracy [30]. FNNs have been proven as a reliable framework for image classification [31].

However, the main difficulty in applying an FNN for hyperspectral classification problems is twofold. First, a large number of weight parameters is required to be learned, proportionally to $Q \prod_{l=1}^{D} p_l + QC$, where variable Q refers to the number of hidden neurons of the network. This outcome derives as a consequence of the structure of the network as is briefly described in the following (see Section IV-A). This, in the sequel, implies that a large number of labeled samples are needed to successfully train the network. Second, the weights of the network are not directly related to the physical properties of hyperspectral data and how these properties affect the classification performance, since the inputs are vectorized and thus they do not preserve their structure. Networks are often treated as "black boxes" when they are applied for the classification of hyperspectral data. To overcome these problems, we propose a modification of conventional FNN structures so that network weights from the input to the hidden layer satisfy the rank-1 canonical decomposition according to statements of Section III. In addition, we introduce a learning algorithm to train the so-called rank-1 canonical decomposition FNN—rank-1 FNN. Before proposing rank-1 FNN, we briefly describe how $p_w^k(\cdot)$ is modeled through an FNN.

A. Feedforward Neural Network Modeling

Fig. 1 illustrates an FNN nonlinearly approximating the probability $p_w^k(\cdot)$. Initially, the tensor input $\boldsymbol{X}$ is vectorized

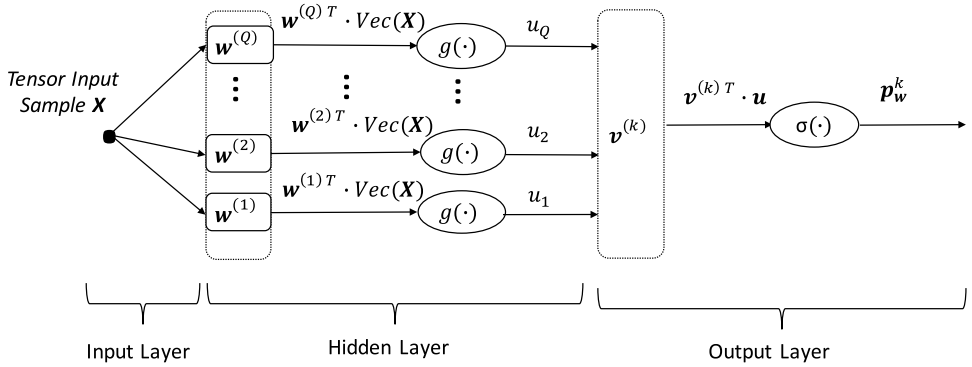

Fig. 1. Structure of an FNN.

creating a vector, vec($\boldsymbol{X}$), of size $\prod_{l=1}^{D} p_l$. The network is assumed to have Q hidden neurons. Each neuron is associated with an activation function $g(\cdot)$. In this paper, the sigmoid function $g(x) = 1/(1+\exp(-x))$ is selected. The activation function of the ith out of Q hidden neurons receives as input the inner product of vec($\boldsymbol{X}$) and a weight vector $\boldsymbol{w}^{(i)}$ associated with the ith neuron and produces as output a scalar u_i given by [32]

$$u_i = g(\boldsymbol{w}^{(i)T}\text{vec}(X)) \equiv g(\langle \boldsymbol{w}^{(i)}, \text{vec}(X)\rangle) \tag{19}$$

where we recall that the operator $\langle\cdot,\cdot\rangle$ expresses the inner product. It should be mentioned that in the current notation, superscript i of the weights $\boldsymbol{w}^{(i)}$ refers to the ith hidden neuron. Gathering the responses of all hidden neurons in one vector $\boldsymbol{u} = [u_1, u_2, \ldots, u_Q]^T$, we have that

$$\boldsymbol{u} = g(\langle \boldsymbol{W}, \boldsymbol{X}\rangle) \tag{20}$$

where $\boldsymbol{W} = [\boldsymbol{w}^{(1)}, \ldots, \boldsymbol{w}^{(Q)}]^T$ is a matrix containing the weights $\boldsymbol{w}^{(i)}$ for all hidden neurons, $i = 1, \ldots, Q$. Thus, the output of the network is given as

$$\boldsymbol{p}_w^k = \sigma\left(\left\langle \boldsymbol{v}^{(k)}, \boldsymbol{u}\right\rangle\right) \equiv \sigma\left(\boldsymbol{v}^{(k)T}\boldsymbol{u}\right) \tag{21}$$

where $\sigma(\cdot)$ stands for the softmax function, and $\boldsymbol{v}^{(k)}$ stands for the weights between the hidden and the output layer and the superscript of the weights for the kth class. The softmax function corresponds to the following conditional probability and is defined as follows for a class i:

$$P(y = i|x) = \frac{\exp(\boldsymbol{x}^T \cdot \boldsymbol{w}_i)}{\sum_{j=1}^{j=C}\exp(\boldsymbol{x}^T \cdot \boldsymbol{w}_j)}. \tag{22}$$

B. Rank-1 Canonical Decomposition Feedforward Neural Networks—Rank-1 FNN

To reduce the number of parameters of the network and to relate the classification results to the spatial and spectral properties of hyperspectral input data, we rank-1 canonically decomposed the weight parameters $\boldsymbol{w}^{(i)}$ as in (10). We should stress that vector $\boldsymbol{w}_l^{(i)}$ relates the input tensor data with the ith hidden neuron and therefore $i = 1, 2, \ldots, Q$.

Then, taking into account the properties of (15), the output of the ith hidden neuron u_i can be written as

$$\begin{aligned} u_i &= g(\langle \boldsymbol{w}^{(i)}, \boldsymbol{X}\rangle) \\ &= g\left(\left\langle \boldsymbol{w}_D^{(i)} \otimes \cdots \otimes \boldsymbol{w}_1^{(i)}, \boldsymbol{X}\right\rangle\right) \\ &= g\left(\left\langle \boldsymbol{w}_D^{(i)} \odot \cdots \odot \boldsymbol{w}_1^{(i)}, \boldsymbol{X}\right\rangle\right) \\ &= g\left(\left\langle \boldsymbol{w}_l^{(i)}, \boldsymbol{\tau}_{\neq l}^{(i)}\right\rangle\right). \end{aligned} \tag{23}$$

In (23), vector $\boldsymbol{\tau}_{\neq l}^{(i)}$ is a transformed version of tensor input $\boldsymbol{X}$. Vector $\boldsymbol{\tau}_{\neq l}^{(i)}$ is independent of $\boldsymbol{w}_l^{(i)}$.That is

$$\tau_{\neq l}^{(i)} = \boldsymbol{X}_{(l)}\left(\boldsymbol{w}_D^{(i)} \odot \cdots \odot \boldsymbol{w}_{l+1}^{(i)} \odot \boldsymbol{w}_{l-1}^{(i)} \odot \cdots \odot \boldsymbol{w}_1^{(i)}\right) \tag{24}$$

where we recall that $\boldsymbol{X}_{(l)}$ is the mode-l matricization of $\boldsymbol{X}$.

Equation (23) actually resembles the operation of a single perceptron with inputs the weights $\boldsymbol{w}_l^{(i)}$ and the transformed version $\tau_{\neq l}$ of the input data. That is, if the rank-1 canonically decomposed weights $\boldsymbol{w}_r^{(i)}$ with $r \neq l$ are known, then $\tau_{\neq l}^{(i)}$ will also be known. Fig. 2 shows the structure of the proposed rank-1 FNN. The model consists of one input layer, one hidden, and one output layer. The main modification of this structure compared with a conventional FNN is in the hidden layer. More specifically, the weights of a hidden neuron are first decomposed into D canonical factors, each expresses the spatial and spectral band contribution to the classification performance. In this figure, we have shaded the weight vector $\boldsymbol{w}_l^{(i)}$, since all the rest weight vectors are used to estimate the transformed input $\tau_{\neq l}^{(i)}$.

C. Learning Algorithm

To train the proposed rank-1 FNN model, the set $\mathcal{S} = \{(\boldsymbol{X}_i, \boldsymbol{t}_i)\}_{i=1}^N$ containing N samples is used. The learning algorithm minimizes the negative log-likelihood [see relation (13)] with respect to network responses $\boldsymbol{y}_i = [\cdots y_{i,k} \cdots]^T$, with $y_{i,k} \equiv p_w^k(\boldsymbol{X}_i)$ and targets $\boldsymbol{t}_i$ over all training samples.

In case of using a conventional NN training algorithm, the estimated weights do not satisfy the rank-1 canonical decomposition assumption expressed by (10). For this reason, in the proposed learning algorithm, we initially fix all the weights $\boldsymbol{w}_r^{(i)}$ with $r \neq l$. This way, we are able to estimate the transformed input vector $\boldsymbol{\tau}_{\neq l}^{(i)}$. Therefore, the only

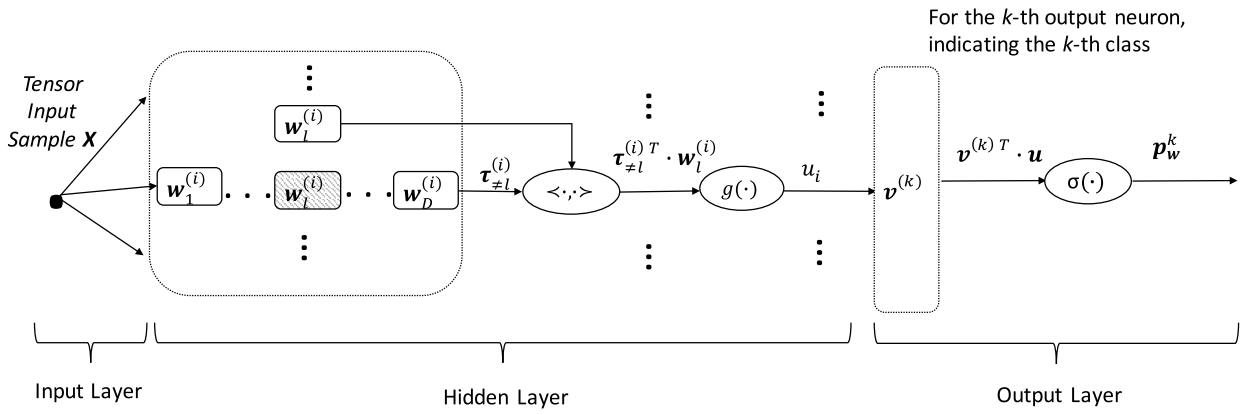

Fig. 2. Structure of the proposed rank-1 FNN.

unknown parameters of the hidden layer is vector $\boldsymbol{w}_l^{(i)}$. This can be derived through a gradient-based optimization algorithm, assuming that the derivative $\partial L/\partial \boldsymbol{w}_l^{(i)}$ is known. Then, the weights are updated toward the negative direction of the partial derivative.

One way to estimate the partial derivative $\partial L/\partial \boldsymbol{w}_l^{(i)}$ is to exploit the principles of the backpropagation algorithm which actually implements the chain rule property for estimating the derivative of the error with respect to the network weights. In particular, by using the backpropagation algorithm, we compute the partial derivatives

$$\begin{aligned} &\partial L/\partial \boldsymbol{w}_l^{(i)} \quad \text{for} \;\; l = 1, \ldots, D \quad \text{and} \;\; i = 1, \ldots, Q \\ &\partial L/\partial \boldsymbol{v}^{(k)} \quad \text{for} \;\; k = 1, \ldots, C. \end{aligned} \tag{25}$$

The partial derivative $\partial L/\partial \boldsymbol{w}_l^{(i)}$ can be estimated if we assume all the weights $\boldsymbol{w}_r^{(i)}$ with $r \neq l$ fixed, since in this case, we can estimate the vector $\boldsymbol{\tau}_{\neq l}^{(i)}$. Therefore, the estimation of the parameters of the rank-1 FNN is obtained by iteratively solving with respect to one of the D canonical decomposed weight vectors, assuming the remaining fixed. Algorithm 2 presents the main steps of the proposed algorithm, applying for the calculation of the hidden layer weights of the network.

Algorithm 2 Optimal Estimation of Hidden Layer Model Weights of the Rank-1 FNN

Initialization:
1. Set Iteration Index $n \rightarrow 0$
2. Randomize all the weight vectors $\boldsymbol{w}_l^{(i)}(n) \in \mathbb{R}^{p_l \times 1}$ for all $l = 1, \ldots, D$ and for all $i = 1, 2, \ldots, Q$
3. Randomize all the weight vectors $\boldsymbol{v}^{(k)}(n) \in \mathbb{R}^{Q \times 1}$ for all $k = 1, \ldots, C$
4. **repeat**
  for $l = 1, \ldots, D$ **do**
    for $i = 1, \ldots Q$ **do**
      4.1 Estimate the transformed input vector $\boldsymbol{\tau}_{\neq l}^{(i)}$ $\tau_{\neq l}^{(i)} = \boldsymbol{X}_{(l)}(\boldsymbol{w}_D^{(i)}(n) \odot \cdots \odot \boldsymbol{w}_{l+1}^{(i)}(n) \odot \boldsymbol{w}_{l-1}^{(i)}(n+1) \odot \cdots \odot \boldsymbol{w}_1^{(i)}(n+1))$,
      4.2 Estimate the derivative error of $\partial L/\partial \boldsymbol{w}_l^{(i)}$
      4.3 Update the weights $\boldsymbol{w}_l^{(i)}(n)$ toward the negative direction of derivative
    end
  end
  for $k = 1, \ldots, C$ **do**
    4.4 Estimate the derivative error of $\partial E/\partial \boldsymbol{v}^{(k)}$
    4.5 Update the weights $\boldsymbol{v}^{(k)}(n)$ toward the negative direction of derivative
  end
  Set $n \rightarrow n + 1$
until *termination criteria are met*;

V. Experimental Results on Hyperspectral Data

A natural question that arises is whether such a reduction in the number of parameters would limit the descriptive power of the models in (9) and in (21). To answer this question, we present quantitative results regarding classification accuracy. Furthermore, we compare the performance of the linear high-order model (tensor-based logistic regression) against two other well-known linear models, vector-based logistic regression and linear SVMs, and the performance of the high-order nonlinear model against fully connected FNNs (FCFFNN), nonlinear SVMs, and two deep learning approaches for classifying hyperspectral data, an approach based on stacked autoencoders (SAEs) [11] and an approach based on the exploitation of convolutional NNs (CNNs) [12]. These methods are the current state of the art in the literature.

The model in (9) is linear in the feature space and the estimated parameters can be used to estimate the importance of the spectral bands. In particular, the most important feature elements should have the highest, in absolute value, weight coefficients, while feature elements uncorrelated with the output variables should have coefficient values close to zero. This way, we can reduce the dimensionality of raw data keeping feature elements with the highest weight coefficients.

A. Data Sets

In this paper, we used airborne visible/infrared imaging spectrometer and reflective optics system imaging spectrometer sensors hyperspectral data sets.[2] In particular, we used:

[2]The code for the rank-1 FNN is available at https://bit.ly/2GRMd85.

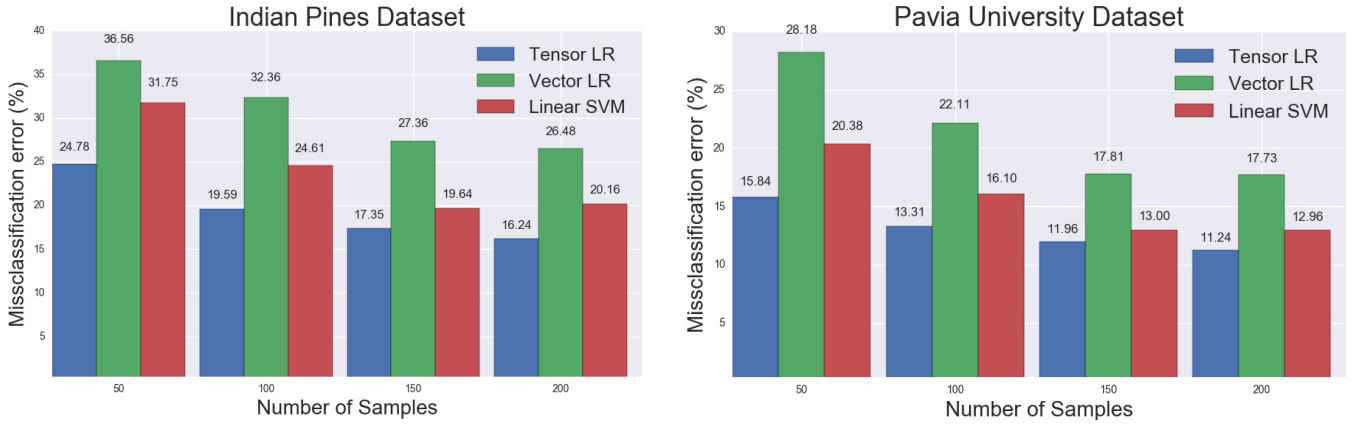

Fig. 3. Misclassification error on test set for tensor-based logistic regression, vector-based logistic regression, and linear SVMs.

1) the Indian Pines data set, which depicts a test site in Northwestern Indiana and consists of 224 spectral reflectance bands and 2) the Pavia University data set, whose number of spectral bands are 103.

A hyperspectral image is represented as a 3-D tensor of dimensions $p_1 \times p_1 \times p_3$, where p_1 and p_2 correspond to the height and width of the image and p_3 to the spectral bands. In order to classify a pixel $I_{x,y}$ at location (x, y) on image plane and successfully fuse spectral and spatial information, we use a square patch of size $s \times s$ centered at pixel $I_{x,y}$. Let us denote as $\boldsymbol{t}_{x,y}$ the class label of the pixel at location (x, y) and as $\boldsymbol{X}_{x,y}$ the tensor patch centered at pixel $I_{x,y}$. Then, we can form a data set $S = \{(\boldsymbol{X}_{x,y}, \boldsymbol{t}_{x,y})\}$ for $x = 1, 2, \ldots, p_2$ and $y = 1, 2, \ldots, p_1$. Each one of the patches $\boldsymbol{X}_{x,y}$ is also a 3-D tensor with dimension $s \times s \times p_3$, which contains spectral and spatial information for the pixel located at (x, y). The data set S is used to train the classifiers. The Pavia University and the Indian Pines data sets contain 42 776 and 10 086 labeled pixels, respectively.

B. Classification Accuracy of the Tensor-Based Logistic Regression

The performance of the tensor-based logistic regression method is evaluated into four different experiments, each of different numbers of training samples to assess classification accuracy even in cases where a small number of samples are used. The evaluation is compared against vector-based logistic regression and linear SVMs. For each of the four experiments, we use as training data set a subset of S that contains 50, 100, 150, and 200 samples from each class, respectively. The remaining samples are used as test data.

The same training data are used to train all models. The vector-based logistic regression as well as the linear SVMs are trained by using the vectorized versions of patches $\boldsymbol{X}_{x,y}$ that belong to the training data sets. The tensor-based logistic regression model is trained by using the same patches without applying any transformation on them and thus keeps intact their spatial structure.

Figure 3 presents the classification accuracy, on the test set, for all tested methods. In both data sets and in all the cases, the tensor-based model outperforms linear SVMs and vector-based logistic regression, despite the fact that it employs the smallest number of parameters. These results quantitatively answer the question regarding the capacity of the model in (9) under a small sample setting framework.

TABLE I

OVERALL CLASSIFICATION ACCURACY RESULTS (%) OF THE TENSOR-BASED LOGISTIC REGRESSION MODEL FOR BOTH DATA SETS

Pavia University			
Splitting ratio	10%	20%	40%
TB-CNN	97.33 ± 0.5	98.41 ± 0.3	99.31 ± 0.2
PCA-CNN	97.58 ± 0.3	98.53 ± 0.3	99.42 ± 0.1
Indian Pines			
Splitting ratio	10%	20%	40%
TB-CNN	82.29 ± 1.1	90.09 ± 0.7	95.98 ± 0.6
PCA-CNN	84.34 ± 0.9	91.72 ± 0.7	96.37 ± 0.5

C. Tensor-Based Dimensionality Reduction

In the following, we evaluate the quality of the dimensionality reduction that can be conducted by the tensor-based logistic regression model. Toward this direction, we utilize the model presented in [12], where a deep CNN is used for classifying hyperspectral data. The authors reduce the dimensionality of the data along the spectral dimension by applying principal component analysis and by utilizing the n principal components that preserve at least 99.9% of the total data set variance.

In this paper, we utilize the same CNN structure, but we reduce the dimensionality of the raw data by selecting the n most significant spectral bands, i.e., spectral bands to which the tensor-based logistic regression model assigns the larger, in terms of absolute value, coefficients. Due to the fact that this method does not take the variation of estimation into account, we first normalized the data, so as to suppress the effect of variance. The results in terms of classification accuracy on the test set, using the same CNN for both dimensionality reduction methods, are presented in Table I. Each experiment has been

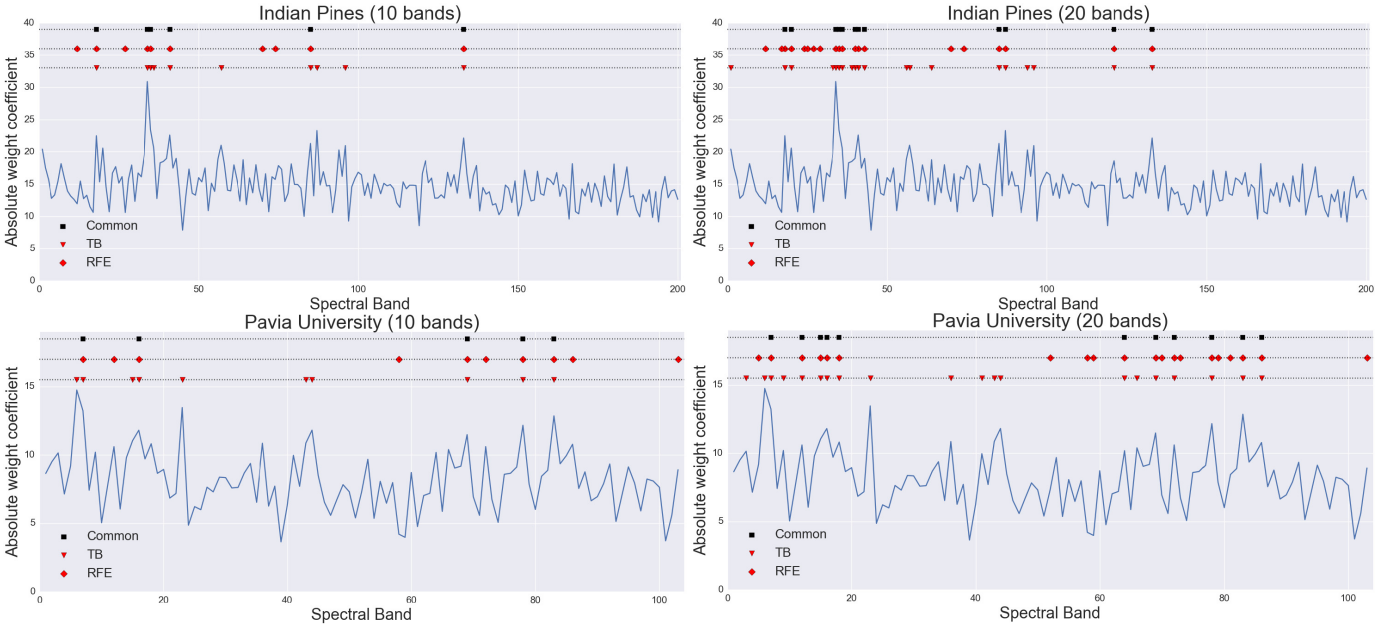

Fig. 4. Selected spectral bands using the TB logistic regression model and the RFE technique. (Left) Selection of 10 most important bands. (Right) Selection of 20 most important bands. Common spectral bands of both techniques are also depicted, denoting as Common in the figure.

executed at 10 different runs and the standard deviation across all different runs is also depicted in Table I. In this table, we denote as TB-CNN the tensor-based (TB) dimensionality reduction followed by a CNN classifier and as PCA-CNN the principle component analysis (PCA) dimensionality reduction followed again by a CNN model. We conducted three experiments where we use 10%, 20%, and 40% of the data sets as the training samples.

The PCA-CNN performs slightly better than the TB-CNN. However, our proposed method presents an advantage over PCA. Although the PCA maps the raw data to a lower dimensional feature space, the resulted features are not interpretable. Using the tensor-based dimensionality reduction, features at the lower dimensional space correspond to the most significant spectral bands providing a physical meaning on how spectral information affects the classification performance.

Furthermore, we compare the features extracted by the proposed tensor-based logistic regression model (see Section III) with the features extracted using the recursive feature elimination (RFE) technique [33]. RFE is an iterative procedure. At each iteration, it trains a model using all available spectral bands, and then, it discards the less informative ones until a predefined number of features are reached. Fig. 4 presents the 10 and 20 most important spectral bands selected from both techniques (i.e., the tensor-based, called TB in the figure and the RFE). In this figure, the common bands detected by both the techniques are also identified for clarification purposes. The results have been conducted for the India Pines and Pavia University data sets. To verify the quality of the selected spectral bands as important, we measure the classification accuracy obtained if we keep only these bands. Specifically, the extracted bands of both techniques are used to train the CNN model presented in [12]. Please note that the training set used for both feature selection techniques (i.e., TB and RFE) is the same and it was built by selecting 100 random samples per class. The results are presented in Table II for Pavia University and Indian Pines data sets. In addition, Fig. 5 shows the confusion matrix for the Pavia data set when 10 or 20 salient bands are selected for the TB and RFE approach. The overall classification accuracy for both dimensionality reduction techniques is almost the same, indicating that the proposed TB method is also appropriate for dimensionality reduction apart from classification capabilities (see Section V-B).

TABLE II

CLASSIFICATION ACCURACY RESULTS (%) WHEN RFE AND TENSOR-BASED LOGISTIC REGRESSION ARE USED TO REDUCE THE DIMENSION OF THE DATA

Feature selection	Number of bands	Accuracy (%) Pavia University	Accuracy (%) Indian Pines
Tensor-based	10	92.82	77.48
RFE	10	92.61	83.42
Tensor-based	20	93.02	77.46
RFE	20	93.25	83.56

The features extracted by the tensor-based logistic regression model are also compared against the features selected by tensor discriminative locality alignment (TDLA) [34], a state-of-the-art method for tensor data dimensionality reduction (see Table III). At this point, we should mention that TDLA has been designed for dimensionality reduction, whereas our method is a tensor-based classification modeling framework. Feature selection is an interesting side effect of our proposed methodology.

Again the quality of the data dimensionality reduction is measured through classification accuracy. In this experiment, we have used different classification models, instead of the CNN adopted previously, to indicate the robustness of the proposed method in selecting salient spectral bands under different classification frameworks. Particularly, the experiments

TABLE III

OVERALL CLASSIFICATION ACCURACY RESULTS (%) WHEN TDLA AND TENSOR-BASED LOGISTIC REGRESSION ARE USED TO REDUCE THE DIMENSION OF THE DATA

DR + Classifier	TDLA + FCFFNN	TDLA + SVM	TB + FCFFNN (10 bands)	TB + SVM (10 bands)	TB + FCFFNN (15 bands)	TB + SVM (15 bands)	TB + FCFFNN (20 bands)	TB + SVM (20 bands)
OA (%)	71.22	89.54	64.53 $\pm$ 1.4	68.14 $\pm$ 0.6	73.28 $\pm$ 0.8	73.77 $\pm$ 0.7	74.62 $\pm$ 0.7	73.95 $\pm$ 0.6

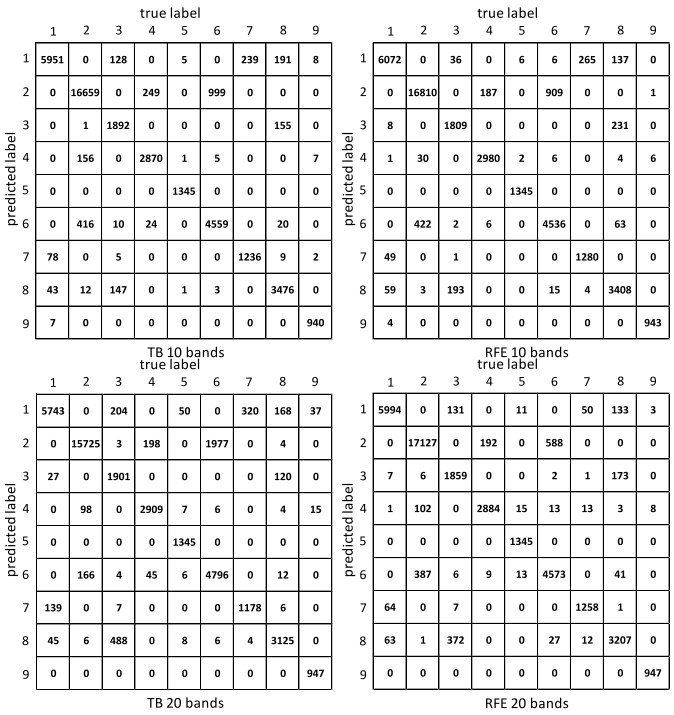

Fig. 5. Classification accuracy per class on Pavia University data set when different feature selection techniques are used.

have been conducted using an FCFFNN and an SVM classifier which receives as inputs the selected salient bands of the TB and TDLA approaches. We have used different classification models (e.g., CNN, FCFFNN, and SVM) to indicate the robustness of the proposed method in selecting salient spectral bands under different classification frameworks. Three different experiments, for 10, 15, and 20 most important spectral bands selection, are conducted. The experiments have been executed using 10 different runs and the standard deviation across all runs is depicted to indicate the robustness of the classification accuracy. As is observed, the best overall classification accuracy is achieved by TDLA and an SVM classifier. However, our approach gives better results when nonlinear models (such as FCFFNN) are adopted.

The conclusions of the aforementioned experiments are that, although the proposed method has been designed for classification purposes (see the respective experiments of Section V-D), it also performs quite well for data dimensionality reduction, compared with state-of-the-art methods that have been appropriately designed for this purpose.

D. High-Order Nonlinear Classification Model

For evaluating the performance of the high-order nonlinear classification model, that is, the rank-1 FNN, we also use the Pavia University and the Indian Pines data sets. A similar procedure as in Section V-B is followed to form the training sets. We conduct different experiments using training sets of 50, 100, 150, and 200 random samples from each class, respectively. A similar training map selection was made in [35] for the Indian Pines and Pavia University data sets. Please note that if for a specific class, the number of samples is less than the number required for the experiment, then we select randomly 50% of the total available class samples. Fig. 6 depicts the effect of different training samples on classification accuracy for different window sizes s and assuming a constant number of neurons in the hidden layer. Particularly, we set $Q = 100$ for the Pavia data set and $Q = 75$ for the Indian Pines. These values are derived from descriptions in the following (see also Fig. 7). In addition, Fig. 7 presents the effect of the number of training samples on misclassification error using different numbers of neurons Q for both data sets.

Then, we evaluate the performance of the proposed tensor-based high-order nonlinear classifier in relation to size s of the spatial window around a pixel. We conduct experiments with a window size to be equal to 3, 5, 7, 9, and 11 as is depicted in Fig. 6. These results suggest that the best overall classification accuracy is achieved when $s = 5$, i.e., the classification accuracy is mostly affected by the 24 closest neighbors of a pixel. When $s < 5$, the spatial information is not adequate for achieving highly accurate classification results, while for $s > 5$, the neighborhood region is very probable to contain pixels that belong to different classes thus deteriorating the classification accuracy.

Moreover, we evaluate the performance of the high-order nonlinear classifier in relation to its complexity, i.e., the value of parameter Q, indicating the number of hidden neurons. Particularly, we set Q to be equal to 50, 75, 100, and 125, respectively. By increasing the value of Q, the complexity and the capacity of the model are also increased. The results of this evaluation are presented in Fig. 7.

Regarding the Indian Pines data set, we observe that the model with $Q = 75$ outperforms all other models. When the training set size is very small, i.e., 50 samples per class, the model with $Q = 50$ does not have the capacity to capture the statistical relation between the input and the output and thus underfits the data. On the other hand, the models with $Q = 100$ and $Q = 125$, due to their high complexity, slightly overfit the data. As training set increases, the misclassification error for all model decreases. This happens because increasing the training set size, it also increases the amount of information that can be used during training to estimate the coefficients of the models.

As far as, the Pavia University data set is concerned, we observe that the model with $Q = 100$ outperforms all other

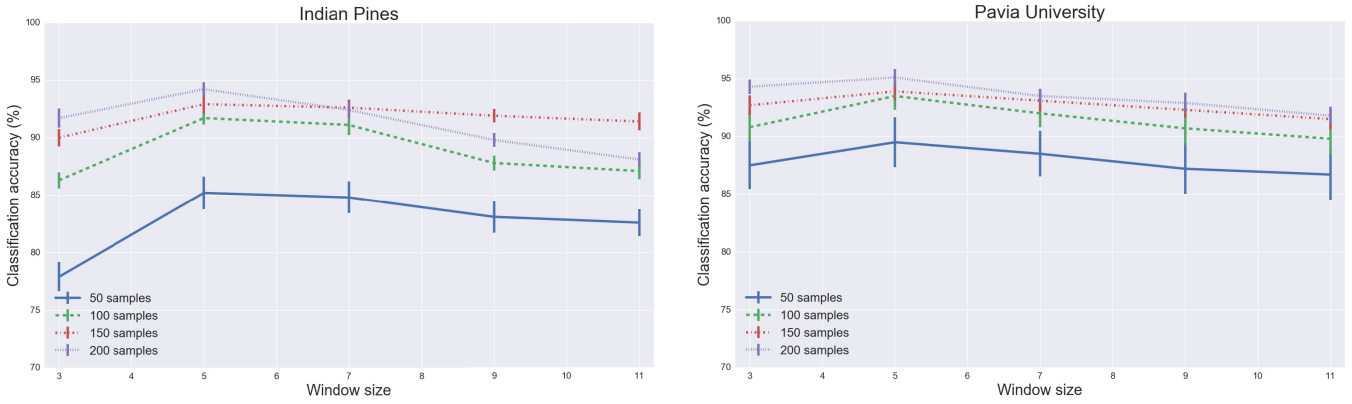

Fig. 6. Overall classification accuracy on test set versus the size of spatial window for the high-order nonlinear classification model.

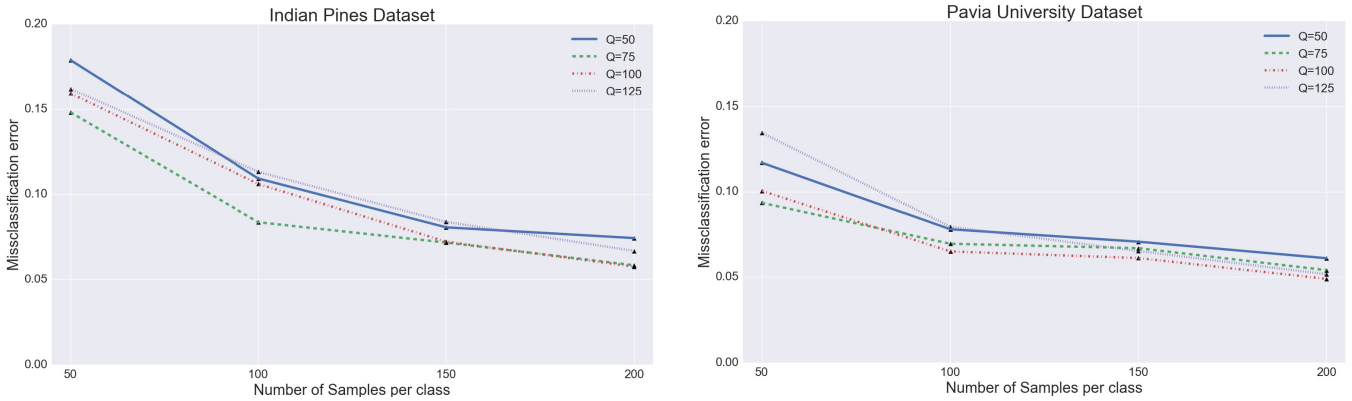

Fig. 7. Misclassification error on test set versus the complexity, determined by Q, of the high-order nonlinear classification model.

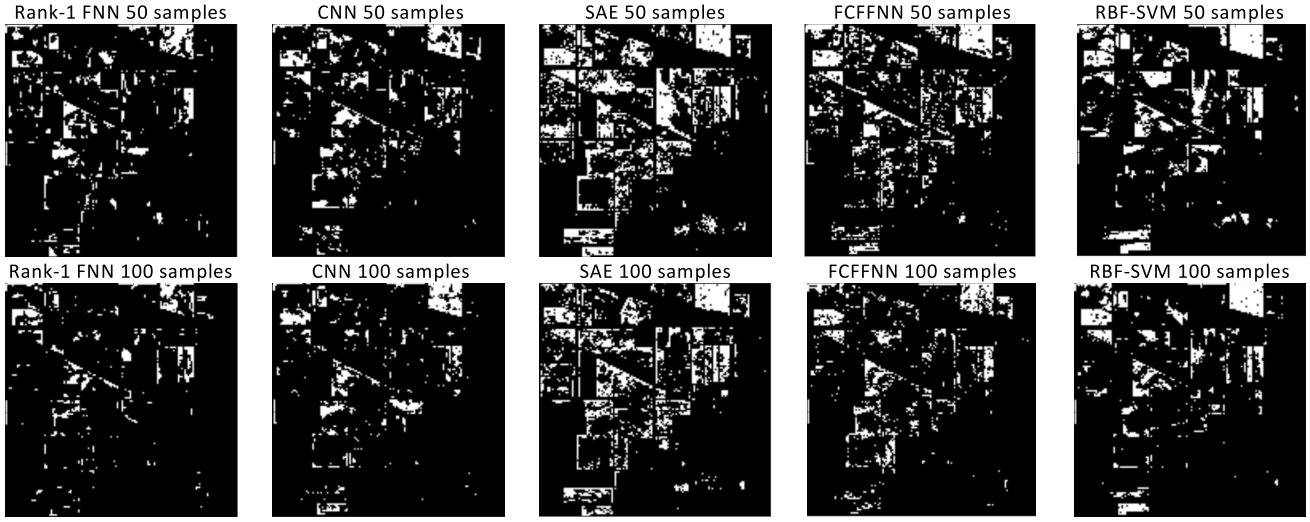

Fig. 8. Visualization of classification accuracy for all tested approaches on Indian Pines data set. White pixels were misclassified.

models, when the size of data set is larger than 50 samples per class. When the training data set size is 50 samples per class, the model with $Q = 75$ outperforms all other models. The model with $Q = 125$ overfits the data, whereas the model with $Q = 50$ underfits them. When the size of the training data set increases, the classification accuracy of all models also increases.

In the following, we compare the performance of the proposed *rank*-1 FNN against the FCFFNN, the radial basis function SVM (RBF-SVM), and two deep learning approaches that have been proposed for classifying hyperspectral data; the first one is based on SAEs [11], whereas the second one is based on the exploitation of CNNs [12]. The FCFFNN receives as input the vectorized version of the rank-1 FNN inputs. The

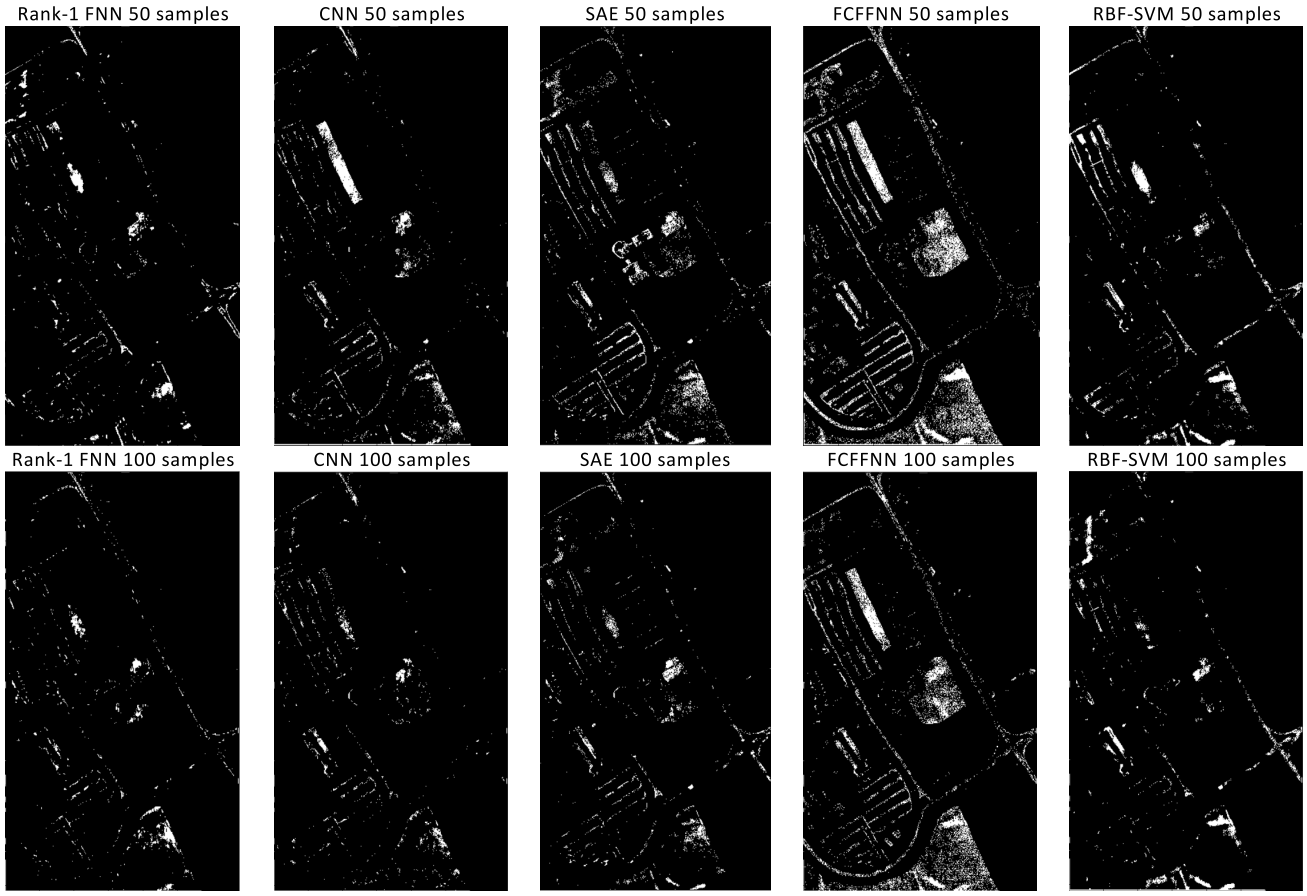

Fig. 9. Visualization of classification accuracy for all tested approaches on Pavia University data set. White pixels were misclassified.

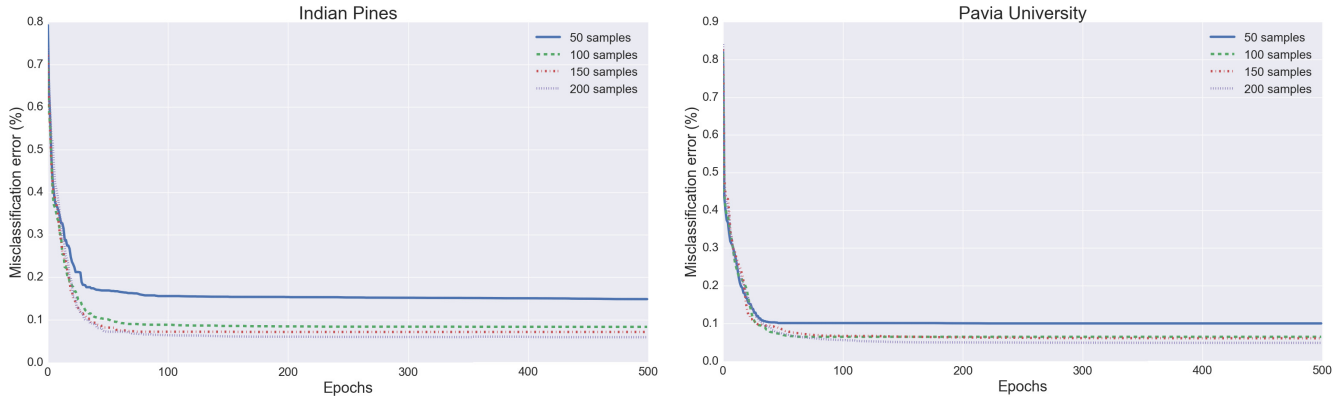

Fig. 10. Convergence curves for the rank-1 FNN for both data sets and for different sizes of training set.

same applies for the input of RBF-SVM. Moreover, FCFFNN has one hidden layer that has the same number of hidden neuron as the rank-1 FNN. The SAE and the CNN were developed as proposed in the original papers. The number of hidden neurons of FCFFNN is 75 for the Indian Pines data set and 100 for the Pavia University data set (as derived from Fig. 7). The architecture of the network that exploits SAEs consists of three hidden layers, while each hidden layer contains 10% less neurons than its input. We choose to gradually reduce the number of hidden neurons from one hidden layer to the next, in order not to permit for the network to learn the identity function during pretraining. Regarding the CNN, we utilize exactly the same architecture as the one presented in [12]. The performance of all these models is evaluated on varying size training sets; training sets contain 50, 100, 150, and 200 samples from each one of the available classes.

Table IV presents the outcome of this comparison. In this table, we also depict the standard deviation across 10 different experiment runs to better reveal the classification accuracy. When the training set size is small, our approach outperforms all other models. This stems from the fact that the proposed

TABLE IV

OVERALL CLASSIFICATION ACCURACY RESULTS (%) OF HIGH-ORDER NONLINEAR CLASSIFICATION MODEL FOR BOTH DATA SETS

Pavia University				
Samples per class	50	100	150	200
rank-1 FNN (Q=100)	**89.95 ± 0.7**	**93.50 ± 0.4**	93.89 ± 0.3	95.11 ± 0.3
FCFFNN	67.79 ± 3.6	76.53 ± 2.7	78.48 ± 2.2	82.59 ± 2.1
RBF-SVM	86.98 ± 1.6	88.99 ± 1.3	89.86 ± 1.2	91.82 ± 1.2
SAE	86.54 ± 2.1	91.90 ± 1.8	92.38 ± 1.4	93.29 ± 1.1
CNN	88.89 ± 0.6	92.74 ± 0.3	**94.68 ± 0.2**	**95.89 ± 0.2**
Indian Pines				
Samples per class	50	100	150	200
rank-1 FNN (Q=75)	**85.20 ± 1.2**	**91.63 ± 0.8**	**92.82 ± 0.5**	94.15 ± 0.4
FCFFNN	73.88 ± 4.2	81.10 ± 3.9	84.14 ± 2.7	85.86 ± 2.5
RBF-SVM	73.18 ± 2.3	77.86 ± 2.0	82.11 ± 1.5	84.99 ± 1.5
SAE	65.51 ± 4.4	70.66 ± 3.6	74.03 ± 3.6	76.49 ± 3.2
CNN	82.43 ± 0.9	85.48 ± 0.7	92.28 ± 0.2	**94.81 ± 0.2**

TABLE V

OVERALL CLASSIFICATION ACCURACY RESULTS (%) OF STM AND RANK-1 FNN FOR BOTH DATA SETS

Indian Pines			
Method	STM	STM-MPCA	Rank-1 FNN
OA	62.6 ± 2.6	80.6 ± 1.9	73.9 ± 1.4
Pavia University			
Method	STM	STM-MPCA	Rank-1 FNN
OA	79.5 ± 1.4	89.4 ± 0.5	88.2 ± 0.8

high-order nonlinear model exploits tensor algebra operations to reduce the number of coefficients that need to be estimated during training, while at the same time, it is able to retain the spatial information of the input. Although the FCFFNN utilizes the same number of hidden neurons as our proposed model, it seems to overfit training sets when a small size of data is used, due to the fact that it employs a larger number of coefficients. The RBF-SVM performs better than the FCFFNN on the Pavia University data set but slightly worse on the Indian Pines data set. The deep learning architecture based on the exploitation of SAE is actually an FCFFNN with three hidden layers, which employs an unsupervised pretraining phase to initialize its coefficients. The fully connectivity property of this architecture implies very high complexity, which is responsible for the poor performance, due to overfitting issues, on the Indian Pines data set. Finally, the deep learning architecture based on the CNN performs better that FCFFNN, RBF-SVM, and SAE mainly due to its sparse connectivity (low complexity) and the fact that it can inherently exploit the spatial information of the input. When the training set consists of 150 and 200 samples per class for the Pavia University data set and 200 samples for the Indian Pines data set, the deep learning architecture based on the CNN seems to outperform even the proposed high-order nonlinear classification model. This happens because the CNN-based deep learning model has higher capacity than the proposed model, which implies that it is capable of capturing better the statistical relation between the input and the output, when the training set contains sufficient information. However, when the size of the training set is small, our proposed model, due to its lower complexity, outperforms the CNN-based deep learning model. Figs. 8 and Fig. 9 depict the classification accuracy for all tested models when the training set contains 50 and 100 samples per class (white labeled pixels correspond to misclassified samples), whereas Fig. 10 presents the convergence curves for the rank-1 FNN. In these figures, the false images (misclassified pixels) are depicted.

Finally, we compare our proposed method against the support tensor machine (STM) proposed in [36] using the Indian Pines and Pavia University data sets. The work of [36] evaluates the performance of STM using both raw data and data of a reduced dimension derived through a multilinear PCA (MPCA) [37]. The results of this comparison are presented in Table V. In this table, we also depict the standard deviation of the results. As is observed, the proposed rank-1 FNN outperforms the STM model when raw hyperspectral data are used. Instead, the MPCA-STM works slightly better than our approach. This is mainly due to the fact that better features are used as inputs.

VI. CONCLUSION

In this paper, we present a linear and a nonlinear tensor-based scheme for hyperspectral data classification and analysis. The advantages of the presented model is that: 1) it reduces the number of parameters required for the classification and thus it reduces the respective number of training samples; 2) it provides a physical interpretation regarding the model coefficients on the classification output; and 3) it retains the spatial and spectral coherency of the input samples. By utilizing tensor algebra operations, we introduce learning algorithms to train the tensor-based classifiers (linear and nonlinear). The linear classifier is based on a modification of a logistic regression model whereas the nonlinear one based on a modification of an FNN. Both the proposed models assume a rank-1 canonical decomposition of the weight parameters. For this reason, we properly modified existing learning schemes to train these models so as to keep the rank-1 canonical decomposition property. We call the new proposed nonlinear classifier as rank-1 FNN.

We have evaluated the performance of the presented model in terms of classification accuracy and for dimensionality reduction. As far as the dimensionality reduction is concerned, the tensor-based scheme allows the selection of the most discriminative spectral bands. It produces an interpretable dimensionality reduction, which can be used with more complicated classifiers without deteriorating their performance. In terms of classification, the linear classifier outperforms conventional linear models, such as logistic regression and SVM.

However, the linear classifier is characterized by low capacity, since it produces classification rules that are linear in the input space. For this reason, in this paper, we have introduced nonlinear classification models the weights of which satisfy the rank-1 canonical decomposition property. We have also introduced suitable learning algorithms to train the new proposed nonlinear model. The performance of the nonlinear model was compared against other nonlinear classifiers, including the state-of-the-art deep learning classifiers. The main results are that when the size of the training set is small, our newly proposed model presents superior performance against the compared methods.

As future work, one can evaluate the performance of the proposed tensor-based classifier on fused data sets that contain hyperspectral data and LiDAR data [38]. In this case, we need to investigate on the extraction of new features that better model how LiDAR data are related with hyperspectral particularities [39].

APPENDIX A

A. Proof of Relation (15)

Proof: For simplicity, we omit the superscripts of $\boldsymbol{w}$. Let us denote as $\boldsymbol{W}$ the tensor

$$\boldsymbol{W} = \boldsymbol{w}_1 \circ \cdots \circ \boldsymbol{w}_D \in \mathbb{R}^{p_1,\ldots,p_D}$$

having the same dimensions with tensor $\boldsymbol{X}$. Then

$$\text{vec}(\boldsymbol{W}) = \boldsymbol{w}_1 \odot \cdots \odot \boldsymbol{w}_D.$$

We also have that

$$\begin{aligned}\langle \boldsymbol{w}_1 \odot \cdots \odot \boldsymbol{w}_D, \boldsymbol{X}\rangle &= \sum_{i_1}^{p_1} \cdots \sum_{i_D}^{p_D} W_{i_1,\ldots,i_D} X_{i_1,\ldots,i_D} \\ &= \langle \boldsymbol{W}_{(d)}, \boldsymbol{X}_{(d)}\end{aligned}$$

for any $d = 1, 2, \ldots, D$. Furthermore, from relation (3) of the main manuscript, we have

$$\begin{aligned}&\langle \boldsymbol{W}_{(d)}, \boldsymbol{X}_{(d)}\rangle \\ &\quad = \langle \boldsymbol{w}_d (\boldsymbol{w}_D \odot \cdots \odot \boldsymbol{w}_{d+1} \odot \boldsymbol{w}_{d-1} \odot \cdots \odot \boldsymbol{w}_1)^T, \boldsymbol{X}_{(d)}\rangle.\end{aligned} \tag{A.1}$$

Now by using (A.1) and the property of inner and dot products, which states that for any three matrices A, B, and C, the following:

$$\langle \boldsymbol{A}\boldsymbol{B}^T, C\rangle = \boldsymbol{A}\boldsymbol{B}^T\boldsymbol{C}^T = \langle \boldsymbol{A}, \boldsymbol{C}\boldsymbol{B}\rangle$$

holds, we conclude that

$$\begin{aligned}&\langle \boldsymbol{w}_1 \odot \cdots \odot \boldsymbol{w}_D, \boldsymbol{X}\rangle \\ &\quad = \langle \boldsymbol{w}_d, \boldsymbol{X}_{(d)} (\boldsymbol{w}_D \odot \cdots \odot \boldsymbol{w}_{d+1} \odot \boldsymbol{w}_{d-1} \odot \cdots \odot \boldsymbol{w}_1)\rangle.\end{aligned}$$

The proof is completed. ◁

REFERENCES

[1] E. Wycoff, T.-H. Chan, K. Jia, W.-K. Ma, and Y. Ma, "A non-negative sparse promoting algorithm for high resolution hyperspectral imaging," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2013, pp. 1409–1413.

[2] C.-I. Chang, *Hyperspectral Data Processing: Algorithm Design and Analysis*. Hoboken, NJ, USA: Wiley, 2013.

[3] G. Camps-Valls and L. Bruzzone, "Kernel-based methods for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 6, pp. 1351–1362, Jun. 2004.

[4] G. Camps-Valls and L. Bruzzone, *Kernel Methods for Remote Sensing Data Analysis*. Hoboken, NJ, USA: Wiley, 2009.

[5] J. M. Benítez, J. L. Castro, and I. Requena, "Are artificial neural networks black boxes?" *IEEE Trans. Neural Netw.*, vol. 8, no. 5, pp. 1156–1164, Sep. 1997.

[6] G. Camps-Valls, D. Tuia, L. Bruzzone, and J. A. Benediktsson, "Advances in hyperspectral image classification: Earth monitoring with statistical learning methods," *IEEE Signal Process. Mag.*, vol. 31, no. 1, pp. 45–54, Jan. 2014.

[7] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[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.

[9] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[10] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 19, 2007, p. 153.

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

[12] K. Makantasis, K. Karantzalos, A. Doulamis, and N. Doulamis, "Deep supervised learning for hyperspectral data classification through convolutional neural networks," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2015, pp. 4959–4962.

[13] M. Vakalopoulou, K. Karantzalos, N. Komodakis, and N. Paragios, "Building detection in very high resolution multispectral data with deep learning features," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2015, pp. 1873–1876.

[14] M. Papadomanolaki, M. Vakalopoulou, S. Zagoruyko, and K. Karantzalos, "Benchmarking deep learning frameworks for the classification of very high resolution satellite multispectral data," *ISPRS Ann. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 3, no. 7, p. 83, 2016.

[15] V. Mnih and G. E. Hinton, "Learning to label aerial images from noisy data," in *Proc. 29th Int. Conf. Mach. Learn. (ICML)*, 2012, pp. 567–574.

[16] K. Makantasis, K. Karantzalos, A. Doulamis, and K. Loupos, "Deep learning-based man-made object detection from hyperspectral data," in *Proc. Int. Symp. Vis. Comput.*, 2015, pp. 717–727.

[17] G. B. Orr and K.-R. Müller, *Neural Networks: Tricks of the Trade*. Berlin, Germany: Springer-Verlag, 2003.

[18] Z. Kandylakis, K. Karantzalos, A. Doulamis, and N. Doulamis, "Multiple object tracking with background estimation in hyperspectral video sequences," in *Proc. IEEE Workshop Hyperspectral Image Signal Process., Evol. Remote Sens. (WHISPERS)*, Jun. 2016, pp. 1–4.

[19] N. Kussul, A. Shelestov, M. Lavreniuk, I. Butko, and S. Skakun, "Deep learning approach for large scale land cover mapping based on remote sensing data fusion," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2016, pp. 198–201.

[20] K. Makantasis, E. Protopapadakis, A. Doulamis, N. Doulamis, and C. Loupos, "Deep convolutional neural networks for efficient vision based tunnel inspection," in *Proc. IEEE Int. Conf. Intell. Comput. Commun. Process. (ICCP)*, Jul. 2016, pp. 335–342.

[21] H. Zhou, L. Li, and H. Zhu, "Tensor regression with applications in neuroimaging data analysis," *J. Amer. Stat. Assoc.*, vol. 108, no. 502, pp. 540–552, 2013.

[22] X. Tan, Y. Zhang, S. Tang, J. Shao, F. Wu, and Y. Zhuang, "Logistic tensor regression for classification," in *Proc. Int. Conf. Intell. Sci. Intell. Data Eng.*, 2012, pp. 573–581.

[23] L. De Lathauwer, B. De Moor, and J. Vandewalle, "Computation of the canonical decomposition by means of a simultaneous generalized schur decomposition," *SIAM J. Matrix Anal. Appl.*, vol. 26, no. 2, pp. 295–327, 2004.

[24] B. C. Csáji, "Approximation with artificial neural networks," M.S. thesis, Faculty Sci., Eötvös Loránd Univ., Budapest, Hungary, 2001, p. 48, vol. 24.
[25] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Rev.*, vol. 51, no. 3, pp. 455–500, 2009.
[26] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.
[27] H. Hung and C.-C. Wang, "Matrix variate logistic regression model with application to EEG data," *Biostatistics*, vol. 14, no. 1, pp. 189–202, 2013.
[28] R. A. Harshman, "Foundations of the PARAFAC procedure: Models and conditions for an 'explanatory' multimodal factor analysis," in *Proc. UCLA Working Papers Phonetics*, 1970, pp. 1–84.
[29] J. D. Carroll and J.-J. Chang, "Analysis of individual differences in multidimensional scaling via an N-way generalization of 'Eckart-Young' decomposition," *Psychometrika*, vol. 35, no. 3, pp. 283–319, 1970.
[30] A. D. Doulamis, N. D. Doulamis, and S. D. Kollias, "On-line retrainable neural networks: Improving the performance of neural networks in image analysis problems," *IEEE Trans. Neural Netw.*, vol. 11, no. 1, pp. 137–155, Jan. 2000.
[31] N. Doulamis, A. Doulamis, K. Ntalianis, and S. Kollias, "An efficient fully unsupervised video object segmentation scheme using an adaptive neural-network classifier architecture," *IEEE Trans. Neural Netw.*, vol. 14, no. 3, pp. 616–630, May 2003.
[32] A. D. Doulamis, N. D. Doulamis, and S. D. Kollias, "An adaptable neural-network model for recursive nonlinear traffic prediction and modeling of MPEG video sources," *IEEE Trans. Neural Netw.*, vol. 14, no. 1, pp. 150–166, Jan. 2003.
[33] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Mach. Learn.*, vol. 46, nos. 1–3, pp. 389–422, 2002.
[34] L. Zhang, L. Zhang, D. Tao, and X. Huang, "Tensor discriminative locality alignment for hyperspectral image spectral–spatial feature extraction," *IEEE Trans. Geosci. Remote Sens.*, vol. 51, no. 1, pp. 242–256, Jan. 2013.
[35] J. Li *et al.*, "Multiple feature learning for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 3, pp. 1592–1606, Mar. 2015.
[36] X. Guo, X. Huang, L. Zhang, L. Zhang, A. Plaza, and J. A. Benediktsson, "Support tensor machines for classification of hyperspectral remote sensing imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 6, pp. 3248–3264, Jun. 2016.
[37] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "MPCA: Multilinear principal component analysis of tensor objects," *IEEE Trans. Neural Netw.*, vol. 19, no. 1, pp. 18–39, Jan. 2008.
[38] F. Pacifici, Q. Du, and S. Prasad, "Report on the 2013 IEEE GRSS data fusion contest: Fusion of hyperspectral and LiDAR data," *IEEE Trans. Geosci. Remote Sens.*, vol. 1, no. 3, pp. 38–136, Sep. 2013.
[39] E. Maltezos, N. Doulamis, A. Doulamis, and C. Ioannidis, "Deep convolutional neural networks for building extraction from orthoimages and dense image matching point clouds," *J. Appl. Remote Sens.*, vol. 11, no. 4, p. 042620, 2017.

Konstantinos Makantasis received the Diploma degree in computer engineering from the Technical University of Crete (TUC), Chania, Greece, the master's degree from the Department of Production Engineering and Management (DPEM), TUC, and the Ph.D. degree with a focus on detection and semantic analysis of object and events through visual cues from DPEM, TUC, in 2016. His diploma thesis was entitled Human Face Detection and Tracking Using AIBO Robots. His master's thesis was entitled Persons Fall Detection Through Visual Cues.

He is mostly involved and interested in computer vision, both for visual spectrum (RGB) and hyperspectral data, and in machine learning/pattern recognition and probabilistic programming. He has been involved for over seven years as a researcher in numerous European and national competing research programs, such as Interreg, FP7, and Marie Curie actions, toward the design, development, and validation of state-of-the-art methodologies and cutting-edge technologies in data analytics and computer vision. He has over 20 publications in international journals and conferences on computer vision, signal and image processing, and machine learning.

Anastasios D. Doulamis received the Diploma degree (Hons.) in electrical and computer engineering and the Ph.D. degree in video analysis from the National Technical University of Athens (NTUA), Athens, Greece, in 1995 and 2001, respectively.

Until 2014, he was an Associate Professor with the Technical University of Crete, Chania, Greece. He is currently a Faculty Member of NTUA. He has authored over 200 papers receiving over 3000 citations.

Dr. Doulamis has received several awards in his studies, including the Best Greek Student Engineer, the Best Graduate Thesis Award, and the National Scholarship Foundation Prize. He has also served as a program committee member of several major conferences of the IEEE and the ACM.

Nikolaos D. Doulamis received the Diploma and Ph.D. degrees (Hons.) in electrical and computer engineering from the National Technical University of Athens (NTUA), Athens, Greece.

He is currently an Assistant Professor with NTUA. He has authored over 55 journal papers and 180 conference papers in the field of signal processing and computational intelligence. He is involved in European projects.

Dr. Doulamis has served as an organizer and/or a program committee member of major IEEE conferences. He has received many awards, including the Best Greek Engineer Student, the Graduate Thesis Award, the NTUAs Best Young Medal, and best paper awards in the IEEE conferences. He has over 3000 citations.

Antonis Nikitakis received the Engineer Diploma degree in electrical and computing engineering from the Democritus University of Thrace, Komotini, Greece, with a specialization in hardware computer and cryptography, the bachelor's degree in psychology from the Psychology Department, University of Crete, Heraklion, Greece, and the master's degree in electronic and computer engineering with a specialization in computer architecture and hardware design, and the Ph.D. degree in electronic and computer engineering with a specialization in the system-on-chip design in computer vision applications from the Technical University of Crete, Chania, Greece.

He has years of experience as a Hardware Engineer involved in the research and the industry. He has also collaborated with a plenty of companies and university to carry out European programs. Moreover, he has collaborated with experimental psychology laboratories and he combines his psychology knowledge with his expertise in computer vision and machine learning. He is currently an Engineer and a Psychologist at Althexis Solutions Ltd., Nicosia, Cyprus.

Dr. Nikitakis received the Best Paper Award at Embedded Systems for Real-Time Multimedia 2012 for his thesis titled High Performance Low Power Embedded Vision Systems.