Capsule Networks for Hyperspectral Image Classification

Mercedes E. Paoletti, *Student Member, IEEE,* Juan M. Haut, *Student Member, IEEE,* Ruben Fernandez-Beltran,

Javier Plaza, *Senior Member, IEEE,* Antonio Plaza, *Fellow, IEEE,* and Filiberto Pla

***Abstract*—Convolutional neural networks (CNNs) have recently exhibited excellent performance in hyperspectral image (HSI) classification tasks. However, the straightforward CNN-based network architecture still finds obstacles when effectively exploiting the relationships between HSI features in the spectral-spatial domain, which is a key factor to deal with the high level of complexity present in remotely sensed HSI data. Despite the fact that deeper architectures try to mitigate these limitations, they also find challenges with the convergence of the network parameters, which eventually limit the classification performance under highly demanding scenarios. In this paper, we propose a new CNN architecture based on spectral-spatial capsule networks in order to achieve highly accurate classification of HSIs while significantly reducing the network design complexity. Specifically, based on Hinton's capsule networks, we develop a CNN model extension which re-defines the concept of capsule units to become spectral-spatial units specialized in classifying remotely sensed HSI data. The proposed model is composed by several building blocks, called spectral-spatial capsules, which are able to learn HSI spectral-spatial features considering their corresponding spatial positions in the scene, their associated spectral signatures, and also their possible transformations. Our experiments, conducted using five well-known HSI datasets and several state-of-the-art classification methods, reveal that our HSI classification approach based on spectral-spatial capsules is able to provide competitive advantages in terms of both classification accuracy and computational time.**

***Index Terms*—Hyperspectral imaging (HSI), Convolutional neural networks (CNN), Capsule networks.**

M. E. Paoletti, J. M. Haut, J. Plaza and A. Plaza are with the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, Escuela Politécnica, University of Extremadura, PC-10003 Cáceres, Spain.(e-mail: mpaoletti@unex.es; juanmariohaut@unex.es; jplaza@unex.es; aplaza@unex.es).

R. Fernandez-Beltran and F. Pla are with the Institute of New Imaging Technologies, University Jaume I, 12071 Castellón, Spain. (e-mail: rufernan@uji.es; pla@uji.es).

I. Introduction

The constant development of spectral imaging acquisition technologies, together with the increasing availability of remote sensing platforms, provide plenty of opportunities to manage detailed spectral-spatial information of the Earth's surface [1]–[3]. As a result, the classification of remotely sensed Hyperspectral images (HSIs) has become one of the most active research fields within the remote sensing community, because it is able to provide highly relevant information for a wide range of Earth monitoring applications such as ecological science [4], [5], precision agriculture [6], [7] and surveillance services [8], among others.

Many different classification paradigms have been successfully adopted by the remote sensing community in order to build effective HSI classifiers [9], [10]. In particular, some of the most noteworthy approaches rely on support vector machines (SVMs) [11], k-means clustering [12], Gaussian process (GP) [13], random forest (RF) [14], extreme learning machines (ELM) [15] and deep neural network classifiers [16]. Despite all the extensive research work conducted in the aforementioned areas, the complex nature of HSI data still makes the classification problem a very challenging one, and also motivates the development of more powerful and accurate classification schemes [17]. Basically, there are two main aspects that HSI classification models need to deal with: (i) high data complexity, and (ii) limited amount of labeled data for training purposes. On the one hand, the high spectral resolution of HSI imaging sensors (typically with hundreds of spectral bands) generates unavoidable signal perturbations as well as spectral redundancies that eventually limit the resulting classification performance. On the other hand, the availability of labeled HSI data for training is usually rather limited, because obtaining accurate ground-truth information is expensive as well as time consuming. This contrasts with the requirement of large amounts of training sets in order to mitigate the so-called Hughes effect [1].

Among all the different HSI classification methodologies presented in the literature, deep learning-based strategies deserve special attention because they have exhibited particularly relevant performance over HSI data due to their potential to effectively characterize spectral-spatial features [18], [19]. From regular stacked auto-encoders (SAE) [20], through sparse auto-encoders (SSA) [21], to deep belief networks (DBN) [22], several kinds of deep learning models have been proposed and successfully adopted to classify HSI data. However, the two-dimensional nature of all these early models typically generates an important spatial information loss, which eventually leads to a limited classification performance (especially under the most challenging scenarios) [23]. Precisely, the most recent approaches try to relieve this constraint by managing the HSI data as a whole three-dimensional volume in order to capture features representing the spectral-spatial domain. For instance, this is the case of the spatial updated deep auto-encoder (SDAE) presented in [24], which improves the regular SAE approach by integrating contextual information. Nonetheless, one of the most relevant improvements was achieved when convolutional neural networks (CNNs) were successfully adapted by Chen *et al.* to classify remotely sensed HSI data [25], achieving the current state-of-the-art performance.

Since Chen *et al.* adopted in [25] the CNN approach for HSI classification purposes, different CNN-based extensions have been also proposed in the literature to learn enhanced spectral-spatial features. For instance, Li *et al.* propose in [26] the use of pixel-pair features under a CNN-based classification scheme in order to increase the number of training samples and, hence, the resulting classification performance. Zhao and Du [27] also propose a classification approach which merges CNN-based spatial features and the spectral information uncovered by the balanced local discriminant embedding algorithm. Other important works make use of several independent CNN-based architectures to combine spectral and spatial features, such as [28], [29]. Despite the fact that all these methods have shown to obtain certain performance benefits, they still struggle at facing the two aforementioned issues when dealing with remotely sensed HSI data, that is, the high data complexity and the limited availability of training samples, mainly because they fuse different data components using independent CNN-based procedures. In this sense, the work presented in [30] defines a novel CNN architecture which is able to jointly uncover improved spectral-spatial features that are useful to classify HSI data.

In general, CNNs have exhibited good performance in

HSI classification due to the fact that convolutional filters provide an excellent tool to detect relevant spectral-spatial features present in the data. That is, initial convolutional layers are able to learn simple HSI features, while deeper layers combine these low-level characteristics to obtain higher-level data representations. However, under this straightforward CNN-based scheme, the capability of exploiting the relationships between features detected at different positions within the image is rather limited. Although the insertion of pooling layers and the gradual reduction of the filters' spatial size allow detecting higher order features in a larger region of the HSI input image (by achieving translation invariance), the internal data representation of a regular CNN does not take into account the existing hierarchies between simple and complex features. Note that the pooling operation is based on downsampling the feature space size to a manageable level and, logically, this introduces an unavoidable loss of information; specifically, pooling methods are unable to capture information about the positional data, which may be a key factor when classifying HSI data. As a result, CNNs may exhibit poor performance if the input data presents rotations, tilts or any other orientation changes, being incapable of identifying the position of one object relative to another in the scene because they cannot model properly and accurately such spatial relationships. Several methods have been implemented in order to encode the invariances and symmetries that exist in the data, including the transformation of the original input samples during the training phase via data augmenting [25], [31]. However this method fails to capture local equivariances in the data, and does not ensure equivariance at every layer within the CNN [32].

Another way to address this problem is to conduct architecture improvements, e.g. by developing deeper networks with a large number of filters. Even though this practice can improve the resulting performance, it requires a significant amount of data to obtain good generalization coupling, which may become an important limitation in some specific scenarios. The rationale behind this effect is based on the vanishing gradient problem [33], which can result in poor propagation of activations and gradients in deep CNNs that ultimately degrades the classification performance. In this sense, the improvements brought to CNN filters (kernels) via the development of residual connections [34], [35] (ResNet) and dense skip connections [36] (DenseNet) open new and interesting paths to uncover highly discriminative spectral-spatial features present in HSI data. On the one hand, the ResNet defines a CNN extension based on processing blocks (residual units [37]), used as fundamental structural entities to allow learning relevant spectral-spatial HSI features from substantially deeper layers. On the other hand, the DenseNet defines an architecture in which each layer concatenates all feature maps coming from the preceding layers as input. Another potential way of encoding complex properties present in the HSI data is defined in [38], where Sabour *et al.* introduced the concept of capsule networks (CapsNets) to encode the data relationships into an activity vector (rather than a scalar) whose length and orientation represents the estimated probability that the object is present and the object's pose parameters, respectively.

With the aforementioned ideas in mind, in this paper we develop a new CNN architecture based on Hinton's CapsNets [38] that achieves highly accurate HSI classification results while significantly reducing the complexity of the network. Specifically, the HSI classification model proposed in this paper is composed by several building blocks, called spectral-spatial capsules, which are able to learn HSI spectral-spatial features considering their corresponding physical positions, their associated spectral signatures, and also their possible transformations. That is, each capsule estimates the probability that a specific spectral-spatial feature is present within the input HSI data and, besides, it provides a set of instantiation parameters that model the transformations suffered by the observed spectral-spatial feature with respect to its corresponding canonical spectral and spatial counterparts. As a result, the proposed network is able to characterize the HSI input data at a higher abstraction level, which eventually allows us to substantially reduce the number of convolutional layers and the inherent model complexity. The proposed network architecture has been accelerated with graphics processing units (GPUs) to further optimize performance. Our experimental results, obtained over five well-known HSI datasets, reveal that the proposed approach exhibits potential to extract highly discriminative spectral-spatial features with a limited amount of training data, providing competitive performance advantages over the spectral-spatial CNN classifier and other relevant state-of-the-art classification methods.

The remainder of the paper is organized as follows. Section II discusses some advantages and limitations of CNNs for HSI classification that motivate the development of our new approach. Section III describes the proposed method. Section IV validates the proposed model by performing comparisons with other state-of-the-art HSI classification approaches over five well-know HSI datasets. Finally, section V concludes the paper with some remarks and hints at plausible future research lines.

II. Advantages and Limitations of CNNs for HSI classification

Let us denote by $\mathbf{X} \in \mathbb{R}^{H \times W \times C}$ a HSI data cube, where H is the height, W is the width and C is the number of spectral bands. Each hyperspectral pixel in $\mathbf{X}$ is a vector of C spectral measures, forming a unique spectral signature for each land-cover material. In deep learning methods, $\mathbf{X}$ can be represented as a vector of $H \cdot W$ elements, where each pixel is denoted as $\mathbf{x}_t \in \mathbb{R}^C$, or as a matrix of $H \times W$ dimensions, where each pixel is described as $\mathbf{x}_{i,j} \in \mathbb{R}^C$, being $i = 1, 2, \cdots, H$, $j = 1, 2, \cdots, W$ and $k = 1, 2, \cdots, H \cdot W$. The relationship between both representations can be expressed as $t = (i-1) \cdot W + j$. This is an interesting point, because traditional standard neural networks are pixel-wise methods that understand the HSI data cube as a list of spectral vectors, for which they define complex, non-linear hypotheses of parameters $\mathbf{W}$ (weights) and B (biases) by applying one or more layers of feature detectors in order to produce the corresponding scalar outputs that summarize the activities of these layers [39].

In this sense, these models assume that each $\mathbf{x}_t$ contains the pure spectral signature of the captured surface material, disregarding the information from surrounding pixels and computing the pixels in isolated fashion [30], [40], [41]. This fact may limit the performance of the classifiers, which becomes strongly dependent on the number ($N_{labeled}$) and quality of the available labeled samples that compose the training dataset $\mathcal{D}_{train} = \{\mathbf{x}_t, y_t\}_{t=1}^{N_{labeled}}$, where y_t is the corresponding category of sample $\mathbf{x}_t$. However, hyperspectral pixels are often highly mixed, introducing high intraclass variability and interclass similarity into $\mathbf{X}$ that is very difficult to avoid, and which often results in characteristic interferences in the obtained classification results (see Fig. 1). Specifically, the CNN model can work as a traditional pixel-wise method, taking each pixel $\mathbf{x}_t$ as an input feature and applying spectral processing (i.e., the so-called 1D-CNN model [25], [30], [41]). However the 1D-CNN cannot always manage the complexity of spectral features, introducing "salt and pepper" noise in the obtained classification (see the leftmost part of Fig. 1). In this sense, it is desirable to incorporate spatial information, i.e. by processing 2D-regions of $\mathbf{X}$, usually centered on pixel $\mathbf{x}_{i,j}$, as input features (i.e., the 2D-CNN model, which exploits the idea that adjacent pixels are intimately related and often belonging to the same class). Combining the information contained in such spatial patches with the spectral signatures (i.e., the 3D-CNN model) can reduce the intra-class variability and improve the final performance. In fact, the potential of CNNs lies in the model architecture, composed by several layers that can be grouped in two well-separated categories: i) the feature extractor net, composed by a stack of layers of artificial neurons (i.e., a convolutional layer followed by a non-linear function and, often, by a subsampling or pooling layer), and ii) the classifier, which can be implemented as a stack of fully connected layers, forming a multilayer perceptron (MLP) or alternatively given by some known technique such as an SVM or LR classifier. The first one obtains high level representations (feature maps), and the second one actually labels the data.

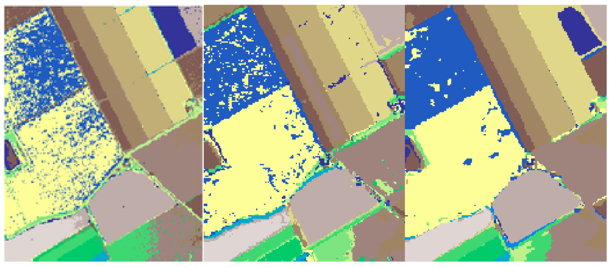


Fig. 1. Characteristic introduced in the classification results obtained by CNN models. Here we show examples of “salt and pepper” noise (1D-CNN, left), misclassified patches (2D-CNN, center) and mixed regions (3D-CNN, right). The examples correspond to extcolorblackan area of a HSI scene collected over the Salinas Valley in California, that will be described in detail in our experimental results section.

Focusing on the feature extractor net, the convolutional layer is the key block of the CNN. Instead of feedforward neural networks (FNNs) such as the MLP, where the group of neurons that compose the l-th layer is fully connected with the neurons of the $l-1$-th and $l+1$-th layers, the l-th convolutional layer is composed by a filter or kernel. The idea behind kernels is related with the statistical properties of images, considered as a stationary source of pixels, where data features are equally distributed into $\mathbf{X}$ in relation to positions [42], suggesting that learned features at one position of $\mathbf{X}$ can be applied to others into $\mathbf{X}$ too, allowing to use the same features at all locations of $\mathbf{X}$. This fact is translated in a convolutional layer by applying its kernel (also called learned feature detector) anywhere in $\mathbf{X}$ in order to obtain a different feature-activation scalar value at each position in the data. In this sense, the l-th layer's kernel is connected and applied over small regions (whose size is defined by the local receptive field) of the input data, called input volume $\mathbf{X}^{(l)}$ (which can be the output volume of the previous layer, i.e. $\mathbf{X}^{(l)} = \mathbf{O}^{(l-1)}$, or the original input image, i.e. $\mathbf{X}^{(l)} = \mathbf{X}$), via local connections and tied weights. This allows reducing the number of connections between layers and, hence, the number of parameters that need to be learned and fine-tuned in the entire CNN. Also, this architecture assumes that elements (such as pixels in a HSI data cube) that are spatially close often belong to the same class, and they collaborate in the task of forming a specific feature of interest, providing additional and valuable information to the classification task and reducing the label uncertainty and intra-class variability due to a better characterization of contextual features. In essence, each kernel of the l-th layer computes the dot product $(\cdot)$ between its own weights $\mathbf{W}^{(l)}$ and a predefined region of the provided input volume to which it is connected as follows:

$$o^{(l)_z}_{i,j,t} = (\mathbf{X}^{(l)} * \mathbf{W}^{(l)})_{i,j,t} = \sum_{\hat{i}=0}^{k-1}\sum_{\hat{j}=0}^{k-1}\sum_{\hat{t}=0}^{q-1} x^{(l)}_{(i\cdot s+\hat{i}),(j\cdot s+\hat{j}),(t\cdot s+\hat{t})} \cdot w^{(l)}_{\hat{i},\hat{j},\hat{t}} + b^{(l)}, \tag{1}$$

where $o^{(l)_z}_{i,j,t}$ corresponds to the (i,j,t) element of the z-th feature map that composes the output volume $\mathbf{O}^{(l)}$ of the l-th convolutional layer, $x^{(l)}_{i,j,t}$ is the (i,j,t) element of the input volume $\mathbf{X}^{(l)}$, $w^{(l)}_{\hat{i},\hat{j},\hat{t}}$ is the $(\hat{i},\hat{j},\hat{t})$ weight of $\mathbf{W}^{(l)}$, $b^{(l)}$ is the bias, and finally s and $k \times k \times q$ are the stride and the kernel size of layer l, respectively. As a result, the obtained $\mathbf{O}^{(l)}$ will be an array of scalar values composed by K 1, 2 or 3-dimensional feature maps, depending on the kernel's dimension.

One mechanism to avoid the degradation that the model can suffer because of the vanishing gradient problem is based on adding a batch normalization layer after the convolutional layer. This kind of layer reduces the covariance shift by means of which the hidden unit values shift around, allowing a more independent learning process. It regularizes and speeds up the training process, imposing a Gaussian distribution on each batch of feature maps as follows:

$$\text{BN}(\mathbf{O}^{(l)}) = \frac{\mathbf{O}^{(l)} - \text{mean}\left[\mathbf{O}^{(l)}\right]}{\sqrt{\text{Var}\left[\mathbf{O}^{(l)}\right] + \epsilon}} \cdot \gamma + \beta, \qquad (2)$$

being γ and β learnable parameter vectors, and ϵ a parameter for numerical stability.

As convolution layers define a linear operation of element-wise matrix multiplication and addition, a detector stage [19] needs to be added after the convolutional and batch normalization layers in order to learn nonlinear representations, composed by a non-linear activation function $\mathbf{O}^{(l)} = \text{f}\left(\mathbf{O}^{(l)}\right)$, where $\text{f}(\cdot)$ defines an element-wise function such as the sigmoid, the tanh or the widely used rectified linear unit (ReLU) [43]–[45]), which computes $\text{f}(\mathbf{O}^{(l)}) = \max(0, \mathbf{O}^{(l)})$, allowing the network to train faster due to its computational efficiency, which also helps to alleviate the vanishing gradient problem without introducing significant differences in the accuracy as compared to other activation functions such as the sigmoid. In this sense, the volume $\mathbf{O}^{(l)}$ will host the neural activations, which is usually interpreted as the likelihood of detecting a certain feature. Those layers closer to the input of the network commonly learn and detect simple features, whereas those layers closer to the output of the CNN combine the previous simple features to learn and detect more complex ones, until combining and learning highly abstract features to produce the final classification.

Finally, following the non-linear activation layers, a downsampling strategy is normally implemented in order to reduce and summarize the dimensionality of each feature map contained in the output volume $\mathbf{O}^{(l)}$ applying a max, average or sum operation (among other recent methods, such as mixed pooling [46], stochastic pooling [47] or wavelet pooling [48]) over a neighborhood window [49]. Non-linear downsampling works independently of the volume's depth, resizing it spatially. For instance, the well-known max pooling examines a window of the output volume $\mathbf{O}^{(l)}$, taking the maximum activation into the region. This working mode reduces the number of parameters, which helps to control overfitting, and provides the network with some kind of invariance to small distortions and transformations that are present in the training data (particularly translation invariance).

Although pooling provides an efficient and simple tool for detecting whether a certain feature is present in any region of the volume $\mathbf{O}^{(l)}$ (looking at the neural activations values), it also implies a certain loss of spatial information concerning the features, which can hamper the classification performance. This effect may lead the CNN model to disregard how different features in the volume $\mathbf{O}^{(l)}$ are related to each other, a piece of information that can be very useful for the final classification results. In such cases, it is common to observe in HSI images that several wrongly classified patches appear near to or even inside well-defined classes, as we can observe in the center and rightmost parts of Fig. 1, where patches belonging to an agricultural field (e.g. the grapes-untrained class in yellow) are misclassified into another class (e.g. the vineyard-untrained class in blue) and vice-versa. These misclassifications are observed in both kinds of models, 2D-CNNs and 3D-CNNs, which indicates that the incorporation of spatial information cannot fully address these problems. This situation could be solved by looking at the logical spatial relationship between both land-cover materials: it seems obvious that, in the case of crop fields, these are arranged in geometric forms, defining clear frontiers between one crop and another. In the case of urban environments, we can also consider how the elements are spatially organized, for example roads could be better defined by assuming that parked cars, sidewalks, ornamental vegetation and buildings will be normally be placed on both sides of the road and not inside. Precisely, the exploitation of this kind of high-level spatial information is one of our main motivations to introduce a new CNN model for HSI remote sensing data classification based on capsules [38], which presents the potential to intelligently exploit both spectral and spatial features from HSI data.

III. Proposed Method

The neural network architecture that we introduce in this work is based on a new convolutional model inspired by the working mode of capsules, with the objective of efficiently preserving the spatial-spectral details of the features present in HSI data cubes and taking advantage from the information obtained at the neuron outputs, which contain vectors of instantiation parameters instead of the classical scalar outputs. Also, in order to provide accurate classification results, our proposal exploits both the spectral and the spatial information contained into the datacube $\mathbf{X}$, implementing a 3D model.

At this point, we emphasize that CNN models have been traditionally employed for remote sensing scene classification, in which the full image $\mathbf{X}$ represents a target. This assumes that the CNN model is fed with a full normalized image prior in order to perform data classification. In our context, we focus on a HSI data cube $\mathbf{X} \in \mathbb{R}^{H \times W \times C}$, which can be understood as a collection of $H \times W$ pixel vectors, and where each pixel $\mathbf{x}_{i,j} \in \mathbb{R}^{C}$ contains the spectral signature of a specific land-cover class (usually highly mixed within the image). That is, each $\mathbf{x}_{i,j}$ represents a target. Our newly proposed neural network model exploits spectral-spatial information, extracting 3D neighboring blocks around each $\mathbf{x}_{i,j}$ (called patches and denoted by $\mathbf{p}_{i,j} \in \mathbb{R}^{d \times d \times C}$), being $d \times d$ the size of the spatial patch and C the number of spectral channels. These patches are labeled with the same category as the central pixel $\mathbf{x}_{i,j}$ and sent to the model as input data, following a border mirroring strategy described in detail in [30].

The proposed architecture is shown in Fig. 2, where two main parts are clearly differentiated. The HSI data introduced into the model is first processed by an encoder network composed by three layers, which works as a feature extractor and classifier. Then, the resulting processed data is introduced into a decoder network, which improves the classification by performing data reconstruction. In the following, we provide the specific details of both parts.

A. Encoder network

Let us first focus on the encoder network, which is located at the beginning of the neural model. This network aims at extracting those relevant features from the HSI data that will help in the classification tasks, providing the most accurate and useful information that increases the reliability of the network. It is composed by three kinds of layers.

1) First layer: The first layer, denoted as $L^{(1)}$, is composed by a classical convolutional layer, which receives the patches $\mathbf{p}_{i,j} \in \mathbb{R}^{d \times d \times C}$ extracted from the original HSI data cube as input features. Its goal is to arrange the HSI data into features that are fed to the

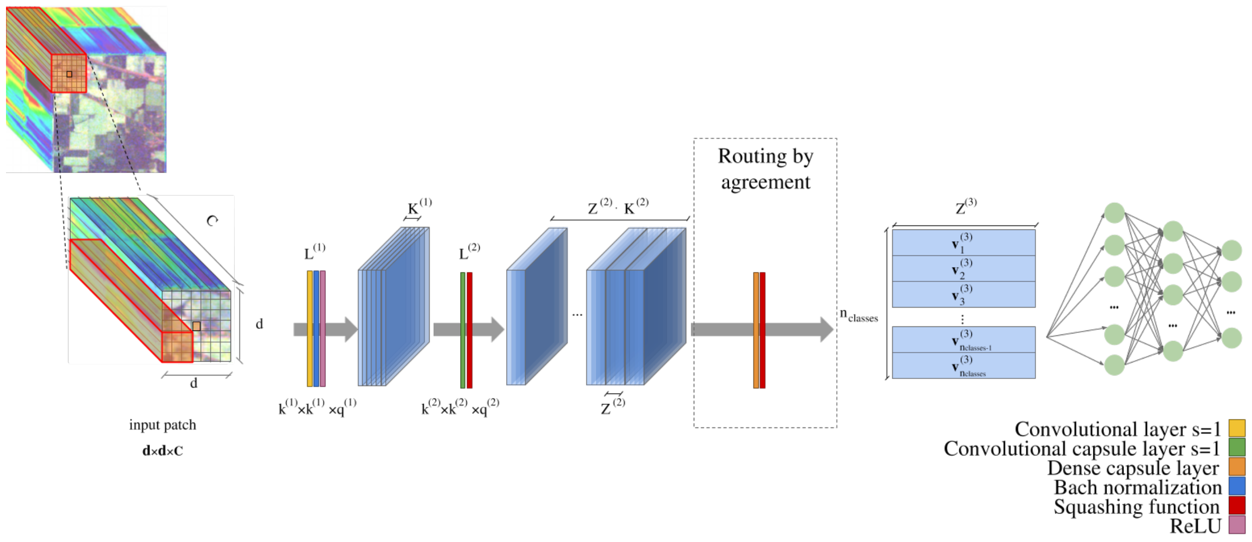


Fig. 2. Proposed neural network architecture. The neural model is composed by an encoder network (in blue) and a decoder network (in green).

subsequent capsule layers, applying a convolution filter of size $k^{(1)} \times k^{(1)} \times q^{(1)}$ (being $q^{(1)} = C$, i.e. it takes into account all the pixel spectrum), followed by a batch normalization step and using the ReLU activation function to obtain an output volume $\mathbf{O}^{(1)} \in \mathbb{R}^{H^{(1)} \times W^{(1)} \times K^{(1)}}$, composed by $K^{(1)}$ feature maps (or channels) of size $H^{(1)} \times W^{(1)}$. This first layer of the encoder prepares the data to obtain the activity vectors of highest capsule-based layers.

2) Second layer: The second layer $L^{(2)}$ (called *primary capsule layer*) can be understood as a matryoshka doll, where $L^{(2)}$ is composed by $K^{(2)}$ convolutional capsules, which in turn are composed by $Z^{(2)}$ convolutional neurons or units with kernel size $k^{(2)} \times k^{(2)} \times q^{(2)}$ (being $q^{(2)} = K^{(1)}$). The working mode is similar to CNN kernels; in fact the m-th capsule will apply its $Z^{(2)}$ units over a region of the volume $\mathbf{O}^{(1)}$, obtaining as a result the output vector $\mathbf{u}_m^{(2)} \in \mathbb{R}^{Z^{(2)}} = [u_{m,1}^{(2)}, u_{m,2}^{(2)}, \cdots, u_{m,Z^{(2)}}^{(2)}]$. These output vectors provide a data structure that is more versatile when storing additional details about the features, such as their orientation, pose or size (in addition to their likelihood), allowing to preserve more detailed information about the spatial relationships observed in the HSI data than standard CNN models. In fact, each element of $\mathbf{u}_m^{(2)}$ represents different properties of the same entity [50]. Here, the concept of entity can be understood as the target object or the object's part of interest (in the HSI domain, the land-cover type) and its associated properties, expressed as the instantiation parameters. In this sense, capsules can be interpreted in the opposite way as rendering in computer graphics, where given an object and its instantiation parameters (such as the pose and the orientation), an image $\mathbf{X}$ is obtained by applying rendering. In our context, the scenario is opposite since, given the image $\mathbf{X}$, the capsule works as an "inverse rendering" unit whose aim is to detect the object and extract the vector of instantiation parameters, called activity vector (see Fig. 3).

In the end, the second layer is performing an inverse rendering process, extracting the lowest level of multi-dimensional entities presnt in the HSI data and grouping them into a 4-D output composed by $K^{(2)}$ feature maps of size $W^{(2)} \times H^{(2)}$, where each element is the activity

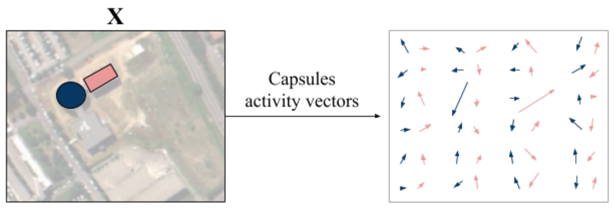


Fig. 3. Given an input image $\mathbf{X}$ with several objects, such as buildings with different shapes, the output of each capsule will be an activity vector whose length and orientation gives the likelihood of the object and its instantiation parameters. In this sense, each capsule is in charge of finding some specific object in $\mathbf{X}$, instead of calculating a feature map (as in the traditional CNN). In this example, focused on an urban area in the University of Pavia scene that will be described later in experiments, the network has 48 capsules, where the black ones try to find buildings with circular shape and the red ones try to find buildings with rectangular shapes.

vector obtained by each capsule of dimension $Z^{(2)}$. An important aspect is that these groups of neurons allow the m-th capsule not only to detect a feature, but also to learn and detect its variants, providing the network with equivariance properties. In that way, the orientation of the m-th capsule's activity vector $\mathbf{u}_m^{(l)}$ in any layer $L^{(l)}$ represents the instantiation parameters, while its length represents the probability that the feature that the capsule is looking for is indeed contained and exists in the input data. In order to properly represent such properties, the length of activity vectors is often scaled down via a non-linear squashing function expressed by Eq. (3), that can be understood as the non-linear activation function of the network model instead of the classical ReLU or sigmoid, for instance, until reaching a magnitude between 0 and 1, leaving their orientation unchanged:

$$\tilde{\mathbf{u}}_m^{(l)} = \frac{\parallel \mathbf{u}_m^{(l)} \parallel^2}{1+ \parallel \mathbf{u}_m^{(l)} \parallel^2} \cdot \frac{\mathbf{u}_m^{(l)}}{\parallel \mathbf{u}_m^{(l)} \parallel}. \quad (3)$$

3) Third layer: After computing the outputs of the primary capsule layer and applying the non-linear squashing function of Eq. (3) over each $\mathbf{u}_m^{(l)}$, the model connects the $K^{(2)}$ capsules in layer $L^{(2)}$ to every capsule in the third layer of the encoder, $L^{(3)}$, denoted as *dense capsule layer*. In this case, $L^{(3)}$ is composed by $n_{classes}$ capsules, which groups $Z^{(3)}$ dense units each one, being $n_{classes}$ the number of different land-cover categories present in the original HSI data cube. For each class, we thus obtain its corresponding activity vector, whose module will encode the probability of each input patch of belonging to that class. In this sense, a special mechanism has been implemented between layers $L^{(2)}$ and $L^{(3)}$, known as *routing-by-agreement* [38], which connects the current dense capsule layer with the previous primary capsule layer. Its goal is to design a better learning process in comparison with traditional pooling methods, not only routing the information between capsules but also capturing part-whole data relationships by reinforcing connections (also understood as contributions) of those capsules allocated at different layers that obtain a high grade of agreement or similarity, while avoiding or deleting the weakest connections. In the following, we provide the details of this mechanism.

The n-th capsule in current layer $L^{(l)}$ takes as input data all the output vectors of the $K^{(l-1)}$ capsules located at previous layer $L^{(l-1)}$, obtaining for each one a prediction vector $\hat{\mathbf{u}}_m^{(l)}$, with $m = 1, 2, \cdots, K^{(l-1)}$, calculated as the weighted multiplication between the m-th capsule's output $\tilde{\mathbf{u}}_m^{(l-1)}$ and the corresponding weights $\mathbf{W}_{m,n}^{(l)}$ (understood as a transformation matrix) that connect the m-th capsule in layer $L^{(l-1)}$ with the n-th capsule in layer $L^{(l)}$, as Eq. (4) shows:

$$\hat{\mathbf{u}}_{n|m}^{(l)} = \mathbf{W}_{m,n}^{(l)} \tilde{\mathbf{u}}_m^{(l-1)} + B_n^{(l)}, \quad (4)$$

where $B_n^{(l)}$ are the biases of capsule n. This equation can be interpreted as a transformation where the output volume from previous primary capsule layer is transformed into $K^{(l)}$ vectors of $Z^{(l)}$ items by applying the transformation matrix $\mathbf{W}_{m,n}^{(l)}$ between the m-th capsule in layer $L^{(l-1)}$ and the n-th capsule in layer $L^{(l)}$.

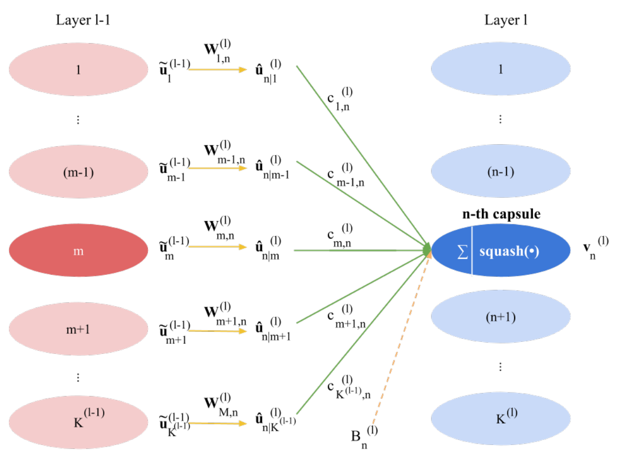

Layer l-1
Layer l
1
(m-1)
m
m+1
$K^{(l-1)}$
$\tilde{\mathbf{u}}_1^{(l-1)}$
$\mathbf{W}_{1,n}^{(l)}$
$\hat{\mathbf{u}}_{n|1}^{(l)}$
$\tilde{\mathbf{u}}_{m-1}^{(l-1)}$
$\mathbf{W}_{m-1,n}^{(l)}$
$\hat{\mathbf{u}}_{n|m-1}^{(l)}$
$\tilde{\mathbf{u}}_{m}^{(l-1)}$
$\mathbf{W}_{m,n}^{(l)}$
$\hat{\mathbf{u}}_{n|m}^{(l)}$
$\tilde{\mathbf{u}}_{m+1}^{(l-1)}$
$\mathbf{W}_{m+1,n}^{(l)}$
$\hat{\mathbf{u}}_{n|m+1}^{(l)}$
$\tilde{\mathbf{u}}_{K^{(l-1)}}^{(l-1)}$
$\mathbf{W}_{M,n}^{(l)}$
$\hat{\mathbf{u}}_{n|K^{(l-1)}}^{(l)}$
$c_{1,n}^{(l)}$
$c_{m-1,n}^{(l)}$
$c_{m,n}^{(l)}$
$c_{m+1,n}^{(l)}$
$c_{K^{(l-1)},n}^{(l)}$
$B_n^{(l)}$
n-th capsule
∑ squash(•)
$\mathbf{v}_n^{(l)}$
1
(n-1)
(n+1)
$K^{(l)}$

Fig. 4. Dynamic routing between capsules: the inferior-layer capsule activity vector is the current input vector $\tilde{\mathbf{u}}_m^{(l-1)}$ of the higher-layer capsule n. After a matrix-transformation given by Eq. (4), $\tilde{\mathbf{u}}_m^{(l-1)}$ is transformed into the prediction vector $\hat{\mathbf{u}}_{n|m}^{(l)}$. The weighted sum [see Eq. (5)] of all the prediction vectors gives as a result the input capsule data $\mathbf{s}_n^{(l)}$ which, after passing through the activation function given by Eq. (6), gives the n-th capsule activity vector $\mathbf{v}_n^{(l)}$.

Moreover, the obtained prediction vectors can be interpreted as the vote of each capsule of $L^{(l-1)}$ in the output of the n-th capsule of $L^{(l)}$, i.e. we can observe each $\hat{\mathbf{u}}_{n|m}^{(l)}$ as a prior prediction of capsule m about the output activity vector of capsule n. This processing allows that capsules at inferior levels can make predictions for capsules at superior levels, increasing the abstraction of the features at each layer. At the end, when multiple predictions agree at different levels, connections between them are strengthened, producing that one higher level capsule will become active for a more complex and abstract feature. This idea of "agreement" is reinforced by introducing, for each prediction vector $\hat{\mathbf{u}}_{n|m}^{(l)}$, a dynamic routing element known as *coupling coefficient* $c_{m,n}^{(l)}$, which relates capsules m and n by calculating the final input $\mathbf{s}_n^{(l)}$ of capsule n as the weighted sum of the previous extcolorblackoutputs of the $K^{(l-1)}$ convolutional capsules in the $L^{(l-1)}$-th layer:

$$\mathbf{s}_n^{(l)} = \sum_{m}^{K^{(l-1)}} c_{m,n}^{(l)} \hat{\mathbf{u}}_{n|m}^{(l)}, \tag{5}$$

which must be squashed by Eq. (3) in order to obtain the final activity vector $\mathbf{v}_n^{(l)}$, whose length represents the probability that the feature target is contained into the data and must be between 0 and 1:

$$\mathbf{v}_n^{(l)} = \frac{\parallel \mathbf{s}_n^{(l)} \parallel^2}{1 + \parallel \mathbf{s}_n^{(l)} \parallel^2} \cdot \frac{\mathbf{s}_n^{(l)}}{\parallel \mathbf{s}_n^{(l)} \parallel}. \tag{6}$$

Focusing again on coupling coefficients, $c_{m,n}^{(l)}$ measures the probability that capsule m activates capsule n, thus all the coupling coefficients of capsule m must sum 1. This parameter is initialized with equal probability for all connections between capsule m in $L^{(l-1)}$ and the $K^{(l)}$ capsules in $L^{(l)}$, and it is obtained by the *routing softmax* expressed by the following equation:

$$c_{m,n}^{(l)} = \frac{\exp\left(b_{m,n}\right)}{\sum_i^{K^{(l)}} \exp\left(b_{m,i}\right)} \text{ with } \sum_i^{K^{(l)}} c_{m,i}^{(l)} = 1, \tag{7}$$

where $b_{m,n}$ denotes the log prior probability that capsule m will activate capsule n, that is, the degree of relationship between both capsules, a measure that is initialized to zero and then refined in each iteration of the network model as follows:

$$\begin{aligned} {}^{(i)}b_{m,n} \leftarrow {}^{(i-1)}b_{m,n} + {}^{(i-1)}a_{m,n} = \\ {}^{(i-1)}b_{m,n} + {}^{(i-1)}\left(\mathbf{v}_n^{(l)} \cdot \hat{\mathbf{u}}_{n|m}^{(l)}\right) = \\ {}^{(i-1)}b_{m,n} + {}^{(i-1)}\left(|\mathbf{v}_n^{(l)}||\hat{\mathbf{u}}_{n|m}^{(l)}|\cos(\theta)\right), \end{aligned} \tag{8}$$

where (i) and $(i-1)$ are the current and previous iterations and ${}^{(i-1)}a_{m,n}$ is the degree of agreement between the prior prediction or vote $\hat{\mathbf{u}}_{n|m}^{(l)}$ and the final output $\mathbf{v}_n^{(l)}$, obtained at iteration $(i-1)$. When $\hat{\mathbf{u}}_{n|m}^{(l)}$ and $\mathbf{v}_n^{(l)}$ are in agreement, we can observe that $\cos(\theta) = \cos(0) = 1$, thus $a_{m,n} = |\mathbf{v}_n^{(l)}||\hat{\mathbf{u}}_{n|m}^{(l)}|$ from a geometrical viewpoint. During the training phase, the network model learns not only the transformation matrices $\mathbf{W}_{m,n}^{(l)}$, encoding the part-hole relationships of the data, but also the coupling coefficients $c_{m,n}^{(l)}$ for each pair of capsules m and n in layers $L^{(l-1)}$ and $L^{(l)}$, respectively. Conceptually, this means that capsules of one layer can make predictions over capsules of the superior layer, grouping those capsules with similar results via dynamic routing in order to obtain clearer outputs, i.e. reinforcing their connections, whereas connections between capsules whose predictions are not related are reduced. Fig. 4 provides a graphical illustration of the dynamic routing process.

We highlight at this point that the main goal of layer $L^{(3)}$, is to obtain as many activity vectors $\mathbf{v}_i^{(l)}$ as the number of objects or land-cover classes present in the image, in such a way that $l = 3$ and $i = 1, 2, \cdots, n_{classes}$). In this sense, for each input data set the proposed neural network obtains a collection of $n_{classes}$ activity vectors, where each $\mathbf{v}_i^{(l)}$ is the capsule for class i, being $\parallel \mathbf{v}_i^{(l)} \parallel$ the probability of belonging to class i. The goodness of the network's output with regards to the desired output can be calculated by the loss function:

$$L_{margin} = \sum_{i}^{n_{classes}} \begin{array}{c} (T_i \max(0, \alpha^+ - \|\mathbf{v}_i^{(l)}\|)^2 + \\ \lambda(1-T_i)\max(0, \|\mathbf{v}_i^{(l)}\| - \alpha^-)^2) \end{array}, \quad (9)$$

where T_i is set to 1 if class i is present in the data extcolorblackand 0 otherwise. We can observe two well-differentiated parts (addends) in Eq. (9). The first one is "activated" when the associated class i is present in the scene (setting $T_i = 1$), while the second one is "activated" in the opposite case, that is, when the associated class i is not present (setting $T_i = 0$). This expression can be extended in order to improve the final classification accuracy by adding a typical reconstruction loss $L_{recon} = \parallel \mathbf{X} - \mathbf{X}' \parallel$, where $\mathbf{X}$ is the original-desired output data and $\mathbf{X}'$ is the network's reconstructed-obtained output data. This reconstruction is performed by the second part of the proposed network, the decoder net, extcolorblackwith the aim of improving the fine-tunning process of the parameters employed in the proposed network.

B. Decoder network

The decoder network is composed by several fully-connected layers that use the output activity vectors of the dense capsule layer to reconstruct the input image, encouraging the capsules to encode the most relevant instantiation parameters of the input data. At the end, the proposed model optimizes the loss function given by Eq. (10) employing the Adam optimizer [51] with learning rate equal to 0.001 and 100 training epochs:

$$L_{final} = L_{margin} + \theta L_{recon}, \quad (10)$$

where θ is a regularization factor to balance the weight between both loss measures that has been fixed to $\theta = 0.0005 \cdot C$ extcolorblackafter a grid search in order to assign an appropriate weight to the reconstruction loss. Also, extcolorblackparameters α^+ and α^- work as boundaries, forcing the length of the activity vector $\|\mathbf{v}_i^{(l)}\|$ (i.e. the probability) in Eq. (9) to lie into a small interval of values in order to avoid maximizing or collapsing the loss. In particular, these boundaries force $\mathbf{v}_i^{(l)}$ to have a length in the range $[0.9, 1]$ if the associated class is present $(\alpha^+ = 0.9)$ and in the range $[0, 0.1]$ in the opposite case. Moreover, $\lambda = 0.5$ works as a regularization parameter to stop the learning, shrinking the impact of those activity vectors whose corresponding classes are not present. Finally, Table I summarizes the layers that compose the proposed model, indicating their configuration parameters, which have been demonstrated a good performance with tested HSI datasets.

IV. Experimental Results

A. Hyperspectral Datasets

Five real hyperspectral datasets have been considered in our experiments (see Table 5). These are the In-

TABLE I

SUMMARY OF THE PARAMETERS IN EACH LAYER OF THE TOPOLOGY OF THE PROPOSED NETWORK.

Input convolutional layer				
Layer ID	*Kernel size* $K^{(i)} \times k^{(i)} \times k^{(i)} \times q^{(i)}$	*Stride*	*Batch normalization*	*Activation function*
$L^{(1)}$	$256 \times 3 \times 3 \times C$	1	Yes	ReLU
Primary capsule layer				
Layer ID	*Kernel size* $Z^{(i)} \times K^{(i)} \times k^{(i)} \times k^{(i)} \times q^{(i)}$	*Stride*	*Activation function*	
$L^{(2)}$	$8 \times 256 \times 3 \times 3 \times 256$	1	Squashing function (eq. 3)	
Dense capsule layer				
Layer ID	*Output size* $n_{classes} \times Z^{(i)}$		*Activation function*	
$L^{(3)}$	$n_{classes} \times 16$		Squashing function (eq. 6)	
Fully-connected layers				
Layer ID	*Number of neurons*		*Activation function*	
$L^{(4)}$	328		Sigmoid	
$L^{(5)}$	192		Sigmoid	
$L^{(6)}$	$d \cdot d \cdot C$		Linear	

dian Pines (IP), Salinas Valley (SV), Kennedy Space Center (KSC) extcolorblackand the full version of the Indian Pines scene, referred hereinafter as the big Indian Pines scene (BIP), all captured by the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) sensor [52], and the University of Pavia (UP) image, acquired by the Reflective Optics System Imaging Spectrometer (ROSIS) sensor [53]. In the following, we provide a description of the aforementioned datasets.

1) *Indian Pines (IP)*: The IP dataset covers an area comprising different agricultural fields in Northwestern Indiana, USA, and it was gathered by the AVIRIS sensor in 1992. This image contains 145×145 pixels with spatial resolution of 20 meters per pixel (mpp) and 224 spectral bands in the wavelength range from 400 to 2500 nm. In our experiments, 4 null bands and other 20 bands corrupted by the atmospheric water absorption effect have been removed. The IP dataset contains a total of 16 mutually exclusive ground-truth classes.
2) *Salinas Valley (SV)*: The SV image was captured in 1998 by the AVIRIS sensor over the Salinas Valley in California, USA. The data comprises 512×217 pixels with spatial resolution of 3.7 mpp. As for the IP dataset, the water absorption bands, i.e. channels from 108th to 112th, from 154th to 167th, together with the 224th band, have been discarded. A total of 16 classes are included in the SV ground-truth data.
3) *Kennedy Space Center (KSC)*: The KSC image was also collected by the AVIRIS instrument (1996) over the Kennedy Space Center in Florida, USA. After removing the noisy bands, the KSC scene contains 176 bands (ranging from 400 to 2500 nm) with 512×614 pixels (20 mpp spatial resolution) and 13 ground-truth classes.
4) *University of Pavia (UP)*: The UP dataset was gathered by the ROSIS sensor (in 2001) over the University of Pavia, Northern Italy. This image contains 103 spectral bands (from 0.43 to 0.86 μm) after several noise-corrupted bands have been discarded, and it comprises 610×340 pixels with 1.3 mpp spatial resolution. The available ground-truth contains 9 different class labels.
5) *Big Indian Pines scene (BIP)*: The BIP image comprises the full flightline of the Indian Pines dataset captured by the AVIRIS sensor in 1992. This image contains 2678×614 pixels (20mpp) and 220 spectral bands ranging from 400 to 2500 nm. The available ground-truth information consists of 58 land-cover categories (some of them spectrally very similar) according to the information provided in Table 5. This dataset is one of the most challenging scenes publicly available to conduct HSI classification due to its considerable size, the very high number of classes, and the imbalanced nature of such classes with very different

numbers of available samples. We emphasize that some classes in the BIP scene have more than 10^4 pixels, but others only contain several tens of samples which poses important challenges for HSI classifiers. As a consequence of the memory restrictions and the large size of this scene, we have reduced the number of spectral bands after applying principal component analysis (PCA) – we retain the first 120 components after PCA–. Although fewer PCA components can explain the variance in the original scene, we have decided to retain a large number of components to illustrate the performance of methods in a challenging scenario from a computational viewpoint.

B. Experimental Settings

A total of extcolorblackeight different classification methods have been selected to conduct the experimental validation in this work. Specifically, the SVM with radial basis function kernel [54], the RF classifier, the multi-layer perceptron (MLP) extcolorblackas well as a deep MLP version with 4 layers (DEEPMLP), the two-dimensional CNN (2D-CNN), the three-dimensional CNN (3D-CNN) [25], the spectral-spatial residual network (SSRN) [35], and the deep fast convolutional neural network (DFCNN) [30] have been compared to the proposed approach. Note that the SVM, RF and MLP are spectral classifiers, while the 2D-CNN is a spatial-based technique and the SSRN, DFCNN (together with the proposed approach) are all spectral-spatial methods. In the case of the 2D-CNN, PCA has been used to reduce the number of HSI bands to a single principal component. Additionally, all the hyper-parameters of the considered methods have been optimally fixed for the experiments.

Regarding the considered classification assessment protocol, three widely used quantitative metrics have been considered to evaluate the classification accuracy: overall accuracy (OA), average accuracy (AA), and Kappa coefficient. All the experiments have been conducted in a hardware environment consisting of a 6th Generation Intel® Core$^{\text{TM}}$i7-6700K processor with 8M of Cache and up to 4.20GHz (4 cores/8 way multi-task processing), 40GB of DDR4 RAM with a serial speed of 2400MHz, an NVIDIA GeForce GTX 1080 GPU with 8GB GDDR5X of video memory and 10 Gbps of memory frequency, a Toshiba DT01ACA HDD with 7200RPM and 2TB of capacity, and an ASUS Z170 pro-gaming motherboard. Regarding our software environment, it is composed by Ubuntu 16.04.4 x64 as operating system, CUDA 9 and cuDNN 7.0.5, PyTorch framework [55] and Python 3.5.2 as programming language.

C. Experiments and Discussion

1) Experiment 1: Our first experiment pursues to validate the performance of the proposed approach with respect to some of the most well-known HSI classification techniques available in the literature. Tables II-V provide a quantitative classification assessment using the IP, UP, SV and BIP datasets, considering the SVM, RF, MLP, 2D-CNN and 3D-CNN classifiers together with the proposed approach. In the tables, class results and global metrics are arranged in rows whereas the considered classifiers are presented in columns. In all these experiments, 15% of the available labeled samples have been used for training, and a spatial size of 11×11 pixels for the input patches was considered for 2D-CNN, 3D-CNN and the proposed method. It should be also mentioned that each table contains the corresponding average and standard deviation values after 5 Monte Carlo runs.

From the results reported on Tables II-V, it is possible to observe that the proposed approach reaches a consistent performance improvement with respect to

TABLE II

CLASSIFICATION RESULTS FOR THE INDIAN PINES (IP) DATASET USING 15% OF THE AVAILABLE LABELED DATA FOR TRAINING AND 11×11 INPUT PATCH SIZE.

Class	SVM	RF	MLP	2D-CNN	3D-CNN	Proposed
1	68.04 ±6.95	33.04 ±7.45	62.39 ±13.96	65.87 ±10.34	89.13 ±7.28	**96.96** ±2.95
2	83.55 ±1.31	66.68 ±1.67	83.84 ±2.46	81.04 ±3.28	98.33 ±0.71	**99.15** ±0.08
3	73.82 ±1.44	56.20 ±2.41	76.37 ±5.03	79.07 ±6.75	98.05 ±1.40	**99.16** ±0.88
4	71.98 ±3.86	41.10 ±2.50	68.35 ±6.12	82.70 ±8.34	98.23 ±0.62	**99.92** ±0.17
5	94.29 ±0.97	87.12 ±1.73	90.87 ±2.09	69.25 ±10.58	97.56 ±2.84	**99.75** ±0.20
6	97.32 ±0.97	95.32 ±1.79	96.95 ±1.10	88.29 ±5.51	98.93 ±1.14	**99.86** ±0.17
7	88.21 ±5.06	32.86 ±12.66	78.21 ±10.28	67.86 ±25.65	83.57 ±19.51	**98.57** ±2.86
8	98.16 ±0.75	98.49 ±0.81	98.08 ±0.90	96.26 ±1.60	99.41 ±0.61	**100.00** ±0.00
9	52.00 ±8.43	13.00 ±3.32	72.00 ±8.12	67.00 ±27.68	65.00 ±21.68	**100.00** ±0.00
10	79.49 ±2.76	69.95 ±4.31	82.17 ±5.41	68.82 ±9.80	97.22 ±0.31	**98.85** ±0.69
11	86.83 ±1.05	90.66 ±1.18	83.66 ±2.85	86.55 ±3.14	98.12 ±2.16	**99.69** ±0.12
12	83.41 ±2.26	55.43 ±4.80	75.89 ±3.33	73.41 ±6.07	93.09 ±5.85	**98.45** ±0.65
13	97.41 ±2.99	93.32 ±2.04	98.68 ±0.54	94.54 ±4.80	99.80 ±0.39	**100.00** ±0.00
14	96.14 ±0.97	96.45 ±0.76	96.17 ±1.02	96.24 ±2.33	99.43 ±0.33	**99.70** ±0.37
15	67.31 ±3.05	50.44 ±2.44	67.80 ±3.56	85.39 ±7.71	96.58 ±2.81	**99.64** ±0.21
16	92.47 ±4.14	85.27 ±3.37	88.71 ±2.77	92.90 ±3.97	93.12 ±3.82	**98.78** ±0.43
OA (%)	86.24 ±0.38	78.55 ±0.68	85.27 ±0.47	83.59 ±0.88	97.81 ±0.56	**99.45** ±0.13
AA (%)	83.15 ±1.10	66.58 ±0.93	82.51 ±1.04	80.95 ±1.55	94.10 ±2.00	**99.34** ±0.40
Kappa	84.27 ±0.45	75.20 ±0.81	83.20 ±0.53	81.23 ±1.04	97.50 ±0.64	**99.37** ±0.14
Time(s)	208.98 ±1.70	1,301.68 ±45.94	**7.31** ±0.15	56.45 ±0.19	39.62 ±0.67	103.21 ±0.47

TABLE III

CLASSIFICATION RESULTS FOR THE UNIVERSITY OF PAVIA (UP) DATASET USING 15% OF THE AVAILABLE LABELED DATA FOR TRAINING AND 11×11 INPUT PATCH SIZE.

Class	SVM	RF	MLP	2D-CNN	3D-CNN	Proposed
1	95.36±0.30	93.52±0.45	94.17±1.73	93.43±2.70	99.16±0.25	**99.99**±0.02
2	98.25±0.16	98.29±0.18	98.06±0.50	97.59±0.88	99.77±0.17	**99.99**±0.01
3	82.93±0.91	75.56±1.86	79.27±7.04	89.96±3.30	96.95±1.78	**99.74**±0.11
4	95.93±0.70	91.68±0.63	94.61±2.58	94.16±3.24	98.80±0.69	**99.82**±0.12
5	99.46±0.36	98.88±0.49	99.63±0.27	97.97±2.69	99.90±0.17	**100.00**±0.00
6	91.76±0.60	74.54±0.97	93.60±1.70	89.62±4.10	99.88±0.12	**100.00**±0.00
7	88.59±0.65	81.01±1.74	88.53±3.47	80.20±4.82	96.54±1.41	**99.64**±0.40
8	90.14±0.54	90.70±0.75	89.59±4.56	96.05±1.88	98.56±0.78	**99.88**±0.07
9	99.97±0.05	99.75±0.26	99.63±0.28	99.48±0.27	99.79±0.19	**100.00**±0.00
OA (%)	95.20±0.13	92.03±0.21	94.82±0.26	94.77±0.72	99.28±0.25	**99.95**±0.02
AA (%)	93.60±0.14	89.33±0.33	93.01±0.60	93.16±1.23	98.81±0.33	**99.90**±0.05
Kappa	93.63±0.17	89.30±0.28	93.13±0.34	93.05±0.97	99.04±0.32	**99.93**±0.03
Time (s)	6.084.92±55.64	6.188.75±35.16	**29.10**±0.92	172.29±0.71	140.09±1.63	471±0.00

SVM, RF, MLP, 2D-CNN and 3D-CNN classification methods, in global sense and also for the individual classes of the IP, UP, SV and BIP datasets. Among all the competitors considered in this initial experiment, the spectral-spatial classifier 3D-CNN obtains the second best result. This is expected, as this method also involves joint spectral-spatial features, which provide more useful information to classify HSI data than the single spectral or spatial features considered by SVM, RF, MLP and 2D-CNN classifiers. Nonetheless, the proposed approach

TABLE IV

CLASSIFICATION RESULTS FOR THE SALINAS VALLEY (SV) DATASET USING 15% OF THE AVAILABLE LABELED DATA FOR TRAINING AND 11×11 INPUT PATCH SIZE.

Class	SVM	RF	MLP	2D-CNN	3D-CNN	Proposed
1	99.68±0.21	99.61±0.12	99.72±0.42	87.99±17.62	100.00±0.00	**100.00**±0.00
2	99.87±0.12	99.86±0.07	99.88±0.15	99.75±0.23	99.99±0.01	**100.00**±0.00
3	99.74±0.11	99.22±0.51	99.43±0.44	81.40±10.85	99.94±0.07	**100.00**±0.00
4	99.48±0.18	99.28±0.44	99.61±0.27	95.11±5.51	99.83±0.23	**100.00**±0.00
5	99.24±0.31	98.46±0.21	99.25±0.48	64.31±12.09	99.90±0.09	**99.99**±0.03
6	99.92±0.06	99.80±0.09	99.92±0.07	99.60±0.11	100.00±0.00	**100.00**±0.00
7	99.70±0.15	99.58±0.09	99.82±0.12	98.01±4.54	99.90±0.15	**100.00**±0.00
8	90.87±0.39	84.41±1.34	85.41±8.00	91.89±2.44	90.67±6.83	**99.53**±0.05
9	99.94±0.02	99.07±0.17	99.86±0.07	98.02±1.56	99.99±0.01	**100.00**±0.00
10	98.26±0.27	93.40±0.58	97.15±0.77	97.05±0.67	99.27±0.43	**99.79**±0.14
11	99.61±0.34	94.79±0.59	97.42±2.29	94.58±3.59	99.48±0.73	**100.00**±0.00
12	99.93±0.05	99.08±0.29	99.80±0.14	92.67±5.75	99.76±0.38	**100.00**±0.00
13	99.07±0.72	98.23±0.69	99.40±0.28	98.10±0.76	99.63±0.58	**100.00**±0.00
14	98.08±1.00	92.81±1.04	97.58±0.94	95.25±5.74	99.94±0.11	**100.00**±0.00
15	72.83±0.78	63.32±1.82	80.27±8.41	87.36±3.87	96.18±1.52	**99.45**±0.23
16	99.45±0.25	98.17±0.36	98.97±0.38	93.72±1.66	99.39±0.42	**99.91**±0.07
OA (%)	94.15±0.10	90.76±0.24	93.87±0.70	92.31±1.62	97.44±1.28	**99.81**±0.03
AA (%)	97.23±0.11	94.94±0.12	97.09±0.33	92.18±2.72	98.99±0.40	**99.92**±0.01
Kappa	93.48±0.11	89.70±0.26	93.18±0.77	91.43±1.81	97.15±1.42	**99.79**±0.03
Time (s)	3.110.30±29.20	4.694.29±158.39	**36.42**±0.11	296.62±3.52	260.41±6.09	1017.40±0.55

is able to consistently outperform the 3D-CNN, with an average improvement of extcolorblack$+1.93$, $+3.84$ and $+2.46$ for OA, AA and Kappa metrics, respectively. extcolorblackAmong all these quantitative results, the experimental comparison conducted over the BIP scene deserves special attention because of the complexity of this dataset. As it can be observed in Table V, the proposed approach obtains the best classification result in all the BIP classes except for Grass/Pasture-mowed and Orchard where it obtains the second best result despite the reduced number of samples of these two classes. Nonetheless, the proposed method achieves a remarkable precision improvement for other small classes, such as, Grass-runway and BareSoil, while also maintaining an important quantitative gain with respect to the other HSI classifiers.

For illustrative purposes, Figs. 6-8 present some of the classification maps corresponding to the experiments reported on Tables II-IV. As it is possible to qualitatively observe in these figures, the classification results obtained by the SVM, RF and MLP techniques tend to be rather noisy, mainly because these methods only

TABLE V

CLASSIFICATION RESULTS FOR THE BIG INDIAN PINES (BIP) DATASET USING 15% OF THE AVAILABLE LABELED DATA FOR TRAINING AND 11×11 INPUT PATCH SIZE.

Class	SVM	RF	MLP	DEEPMLP	2D-CNN	3D-CNN	Proposed
1	16.37±0.83	15.79±0.00	24.56±4.71	21.75±3.94	60.35±11.46	36.14±20.48	**75.44**±5.26
2	59.21±0.42	60.19±0.86	55.39±1.97	64.21±1.21	74.07±1.08	95.00±1.04	**95.87**±0.54
3	79.71±7.20	83.48±2.69	96.81±1.92	97.39±1.92	83.48±10.87	76.81±34.28	**100.00**±0.00
4	39.82±0.34	50.31±0.45	42.65±2.98	59.15±1.65	72.42±1.37	95.03±1.28	**98.04**±0.20
5	21.52±1.37	15.19±0.00	45.57±4.65	48.73±5.88	66.20±9.60	66.96±31.31	**87.34**±7.59
6	26.46±0.57	18.87±0.90	46.19±4.66	47.55±5.22	71.67±3.26	74.86±14.67	**95.33**±0.00
7	25.04±0.33	35.68±1.36	40.19±3.18	54.97±2.43	70.95±2.89	91.03±2.68	**95.82**±0.11
8	32.45±0.34	49.34±0.42	39.54±3.52	61.54±3.48	66.78±2.88	92.39±3.32	**97.58**±0.08
9	50.05±0.40	69.53±0.20	58.17±4.90	74.39±1.17	70.71±1.57	95.25±1.73	**98.72**±0.36
10	68.55±0.49	85.39±0.19	57.50±4.02	76.29±1.63	83.64±0.99	96.76±1.39	**99.06**±0.34
11	41.63±0.41	19.38±1.22	53.78±4.44	70.98±3.36	50.50±8.04	97.55±1.54	**98.81**±1.19
12	17.63±1.37	25.37±1.04	41.93±2.39	50.51±3.03	67.62±5.93	90.60±7.27	**95.34**±2.23
13	55.64±1.13	50.85±1.62	72.14±2.32	81.60±2.42	75.73±2.99	95.25±2.15	**98.00**±1.62
14	31.80±0.13	46.64±0.49	44.44±3.57	67.22±1.21	72.02±1.49	94.93±3.06	**98.44**±0.73
15	52.06±0.67	61.14±0.57	57.12±1.77	77.33±1.11	70.61±2.98	94.42±2.05	**98.94**±0.43
16	49.45±0.81	60.79±1.38	57.60±2.79	72.74±2.97	81.03±1.37	93.17±3.07	**98.35**±0.45
17	40.13±0.47	40.35±1.77	58.24±2.00	61.00±3.96	73.45±3.84	93.85±1.66	**98.05**±1.04
18	63.38±0.69	69.11±0.76	70.05±1.76	80.82±1.21	73.84±3.02	92.41±10.74	**98.87**±0.18
19	30.99±5.42	16.67±1.75	65.79±4.26	69.12±8.12	94.39±5.10	82.11±11.96	**100.00**±0.00
20	29.38±2.33	27.85±1.81	37.79±1.86	41.97±4.16	69.56±2.37	78.52±5.72	**88.27**±0.92
21	65.55±1.03	69.50±0.78	71.55±3.06	78.99±1.70	84.22±3.74	93.62±3.93	**97.51**±0.00
22	15.79±0.00	15.79±0.00	61.05±9.76	57.89±17.30	62.11±9.06	17.89±18.71	**68.42**±10.53
23	15.53±0.65	15.07±0.00	34.79±5.52	29.59±3.93	48.49±6.80	64.38±20.61	**75.34**±0.00
24	18.02±2.55	16.76±1.08	49.73±3.24	42.16±13.95	**62.16**±9.36	9.73±11.67	41.89±14.86
25	57.80±0.52	64.96±1.08	62.68±2.74	72.23±1.85	74.02±3.73	80.37±12.60	**92.46**±0.89
26	54.39±0.77	62.27±2.18	69.20±2.17	78.94±1.79	74.95±2.25	90.09±7.60	**98.40**±0.50
27	83.97±1.38	85.21±0.56	86.33±0.67	90.00±1.03	81.05±1.96	96.17±1.83	**99.31**±0.02
28	98.21±0.96	97.23±0.77	98.57±0.66	98.48±1.12	69.38±18.07	98.13±0.95	**99.55**±0.00
29	38.87±0.36	48.10±1.11	46.20±3.44	61.97±2.83	73.54±3.54	87.12±5.16	**92.99**±0.26
30	34.62±0.49	37.03±1.52	45.15±3.82	53.63±2.19	76.73±1.04	81.23±4.74	**92.28**±0.43
31	16.92±0.37	23.40±2.54	30.69±3.63	31.64±4.38	64.54±4.46	61.67±18.66	**74.33**±4.18
32	16.24±1.21	15.38±0.00	31.28±8.49	32.31±6.61	**81.03**±14.29	74.36±25.49	74.36±10.26
33	73.35±0.34	77.18±0.39	69.57±1.90	80.40±1.19	78.47±1.25	95.98±1.43	**98.00**±0.28
34	50.33±14.98	36.67±2.02	59.02±4.74	51.18±9.33	70.39±9.12	49.22±22.90	**75.49**±11.76
35	43.85±0.04	57.58±0.07	48.58±0.72	66.56±1.31	71.41±0.60	91.16±1.06	**96.81**±0.22
36	28.04±0.21	26.51±2.00	50.92±2.57	53.89±3.21	60.74±5.46	87.14±7.79	**98.32**±0.11
37	17.54±0.56	29.59±1.50	41.51±1.57	49.44±4.36	73.06±1.31	90.85±2.59	**94.91**±1.94
38	23.30±0.26	43.95±0.45	42.82±4.41	62.27±1.01	64.33±3.33	93.48±1.65	**98.29**±0.02
39	49.63±1.11	41.03±1.65	51.16±2.86	71.23±1.83	71.80±5.04	93.35±2.35	**98.88**±0.53
40	47.71±0.13	60.18±1.33	51.47±4.00	69.02±2.14	78.94±1.07	95.32±1.27	**98.09**±0.33
41	35.29±0.17	40.30±0.63	39.81±2.08	59.97±2.19	70.53±1.47	94.92±2.20	**97.69**±0.31
42	38.98±1.20	50.71±0.77	52.12±1.98	67.77±2.54	69.77±2.17	93.18±2.52	**97.57**±0.51
43	30.94±0.94	22.58±4.16	59.89±5.22	69.87±3.05	49.21±3.78	94.92±6.55	**99.26**±0.18
44	45.86±0.24	67.80±0.55	55.74±1.76	74.71±1.19	72.30±2.47	95.03±0.81	**98.07**±0.14
45	57.97±0.38	72.73±1.05	65.07±2.22	80.16±2.58	76.27±2.20	91.34±6.30	**98.16**±0.15
46	40.43±0.52	60.50±2.14	60.41±2.23	77.39±3.03	78.00±1.70	94.10±4.29	**98.09**±0.16
47	64.18±0.34	73.37±0.44	69.46±2.02	83.89±0.56	76.05±1.56	97.64±0.25	**98.79**±0.02
48	38.42±0.32	43.96±0.95	55.20±4.26	73.82±1.87	68.29±2.76	96.16±1.24	**98.65**±0.59
49	45.62±0.73	60.92±0.67	59.64±2.50	73.95±1.72	77.29±1.45	92.19±1.63	**98.22**±0.58
50	59.17±0.44	61.40±1.10	66.46±4.00	79.05±2.72	81.82±1.11	94.50±3.07	**98.01**±0.30
51	30.89±1.07	31.43±2.51	45.66±4.60	59.76±5.82	76.53±3.29	82.63±12.54	**97.09**±0.11
52	66.48±0.14	81.76±0.83	68.51±1.68	81.76±0.93	76.18±2.51	95.94±1.20	**98.39**±0.18
53	85.65±2.08	81.27±0.50	89.02±1.80	91.08±2.01	77.15±5.21	97.77±2.09	**99.31**±0.34
54	99.35±0.24	98.87±0.20	98.77±0.31	99.42±0.15	98.93±0.84	99.41±0.97	**99.98**±0.02
55	67.47±2.23	69.31±1.70	77.45±3.57	84.21±0.86	82.69±2.38	93.24±3.82	**99.83**±0.17
56	85.68±0.48	87.71±0.30	83.08±0.96	88.42±0.51	85.40±1.91	95.75±2.43	**97.76**±0.27
57	94.25±0.10	95.89±0.15	92.46±0.83	94.88±0.56	96.87±0.25	98.59±0.36	**99.60**±0.09
58	54.40±4.58	31.25±4.54	82.50±4.42	90.69±4.78	83.47±3.52	86.53±11.16	**90.97**±0.69
OA (%)	60.43±0.02	69.96±0.09	63.06±0.34	76.12±0.15	79.20±0.68	95.21±0.29	**98.25**±0.03
AA (%)	46.93±0.19	50.98±0.20	58.43±0.18	67.96±0.58	73.57±1.19	85.83±1.92	**93.92**±0.78
Kappa	56.89±0.03	67.17±0.10	60.10±0.35	74.14±0.16	77.45±0.75	94.83±0.32	**98.11**±0.03
Time (s)	476.09±7.20	1688.76±13.74	**148.79**±2.32	343.96±3.55	1887.70±1.06	1364.82±16.67	8430.54±48.10

consider the spectral information contained in the HSI data. In addition, the 2D-CNN tends to introduce some artifacts in class boundaries. This is due to the fact that it only considers the spatial information to provide a pixel prediction, which makes the method quite sensitive to the spatial size of the input patches. Regarding the classification maps produced by the spectral-spatial classifiers, we can observe that the 3D-CNN generates better results than the SVM, RF, MLP and 2D-CNN in terms of class consistency. However, the proposed approach produces better results in terms of border delineation and overall accuracy. For instance, looking at Fig. 7 we can see that the classification map produced by the proposed approach [see Fig. 7(h)] exhibits less misclassified pixels than the corresponding map generated by the 3D-CNN [see Fig. 7(g)]. Another important observation is related to the generalization capability of the proposed approach. Specifically, if we look at the unlabeled image areas (i.e., those that are not covered by the ground-truth), the proposed method appears to provide more consistent classification results (with less potential outliers and artifacts) in those areas than the other considered methods.

2) Experiment 2: In a second experiment, we conduct a specific comparison between the proposed approach and two recent state-of-the-art spectral-spatial HSI classification networks, i.e. SSRN [35] and DFCNN [30]. Table VI compares the proposed approach with the SSRN when considering multiple spatial sizes for the input patches, i.e. 5×5, 7×7, 9×9 and 11×11, using the IP, KSC and UP datasets. Note that the tested spatial sizes are presented in rows and the considered datasets are arranged in columns to show the average OA result and also the corresponding standard deviation in brackets (after 5 Monte Carlo runs). In this experiment, we have selected 20% of the available labeled data for the IP and KSC scenes, and 10% of the available labeled data for the UP scene.

As shown in the results reported on Table VI, the proposed network architecture consistently outperforms the SSRN for most tested configurations. More specifically, the average overall accuracy improvements achieved by the proposed approach are $+2.12$, $+0.51$, $+0.39$ and $+0.45$ for 5×5, 7×7, 9×9 and 11×11 input spatial sizes, and $+2.12$, $+0.25$, $+0.23$ for the IP, KSC and

TABLE VI
OVERALL ACCURACY (%) ACHIEVED BY THE SSRN METHOD [35] AND THE PROPOSED APPROACH WHEN CONSIDERING DIFFERENT SPATIAL SIZES FOR THE INPUT PATCHES.

	Indian Pines (IP)		Kennedy Space Center (KSC)		University of Pavia (UP)	
Spatial Size	**SSRN [35]**	**Proposed**	**SSRN [35]**	**Proposed**	**SSRN [35]**	**Proposed**
5×5	92.83 (0.66)	**97.79** (0.40)	96.99 (0.55)	**97.98** (0.21)	98.72 (0.17)	**99.13** (0.08)
7×7	97.81 (0.34)	**99.30** (0.11)	**99.01** (0.31)	98.85 (0.27)	99.54 (0.11)	**99.75** (0.03)
9×9	98.68 (0.29)	**99.67** (0.06)	99.51 (0.25)	**99.52** (0.16)	99.73 (0.15)	**99.89** (0.02)
11×11	98.70 (0.21)	**99.74** (0.09)	99.57 (0.54)	**99.73** (0.10)	99.79 (0.08)	**99.93** (0.02)

TABLE VII
QUANTITATIVE COMPARISON OF THE 3D-CNN [25], DFCNN [30] AND THE PROPOSED APPROACH WITH THE INDIAN PINES (IP) DATASET USING DIFFERENT SPATIAL SIZES FOR THE INPUT PATCHES.

Class	Samples	3D-CNN [25]	DFCNN [30]			PROPOSED	
		27×27	9×9	19×19	29×29	9×9	15×15
Alfatfa	30	100.00	100.00	100.00	100.00	100.00	100.00
Com-notill	150	96.34	90.57	94.06	**97.17**	96.80	97.11
Com-min	150	99.49	97.69	96.43	98.17	**99.60**	99.48
Corn	100	100.00	99.92	100.00	100.00	100.00	100.00
Grass/Pasture	150	99.91	98.10	98.72	98.76	**100.00**	99.86
Grass/Trees	150	99.75	99.34	99.67	100.00	**100.00**	99.86
Grasslpasture-mowed	20	100.00	100.00	100.00	100.00	100.00	100.00
Hay-windmwed	150	100.00	99.58	99.92	100.00	100.00	100.00
Oats	15	100.00	100.00	100.00	100.00	100.00	100.00
Soybeans-notill	150	98.72	94.28	97.63	99.14	99.31	**99.45**
Soybeans-min	150	95.52	87.75	92.93	94.59	**97.31**	97.24
Soybean-clean	150	99.47	94.81	97.17	99.06	**99.60**	99.55
Wheat	150	100.00	100.00	100.00	100.00	100.00	100.00
Woods	150	99.55	98.09	97.88	99.76	99.45	**99.82**
Bldg-Grass-Tree-Drives	50	99.54	89.79	95.80	98.39	**99.05**	98.62
Stone-steel towers	50	99.34	100.00	99.57	98.92	100.00	99.64
Overall Accuracy (OA)		97.56	93.94	96.29	97.87	98.69	**98.72**
Average Accuracy (AA)		99.23	96.87	98.11	99.00	**99.45**	99.41
Kappa		97.02	93.12	95.78	97.57	98.50	**98.54**

TABLE VIII
QUANTITATIVE COMPARISON OF THE 3D-CNN [25], DFCNN [30] AND THE PROPOSED APPROACH WITH THE UNIVERSITY OF PAVIA (UP) DATASET USING DIFFERENT SPATIAL SIZES FOR THE INPUT PATCHES.

Class	Samples	3D-CNN [25]	DFCNN [30]			PROPOSED	
		27×27	15×15	21×21	27×27	9×9	15×15
Asphalt	548	99.36	97.53	98.80	98.59	99.79	**99.98**
Meadows	540	99.36	98.98	99.46	99.60	99.95	**99.98**
Gravel	392	99.69	98.96	99.59	99.45	98.84	**99.90**
Trees	542	99.63	99.75	99.68	99.57	99.89	**99.97**
Painted metal sheets	256	99.95	99.93	99.78	99.61	100.00	100.00
Bare Soil	532	99.96	99.42	99.93	99.84	99.99	**100.00**
Bitumen	375	100.00	98.71	99.88	100.00	99.97	**100.00**
Self-Blocking Bricks	514	99.65	98.58	99.53	99.67	99.85	**99.87**
Shadows	231	99.38	99.87	99.79	99.83	99.96	**100.00**
Overall Accuracy (OA)		99.54	98.87	99.47	99.48	99.86	**99.97**
Average Accuracy (AA)		99.66	99.08	99.60	99.57	99.81	**99.97**
Kappa		99.41	98.51	99.30	99.32	99.82	**99.96**

UP datasets, respectively. In addition, it is also possible to observe that the standard deviation in the experiments with the proposed method is substantially lower than that in the experiments with the SSRN. This fact, together with the higher OA results, indicates that the proposed architecture is able to effectively reduce the uncertainty when classifying HSI data. The proposed architecture aims at learning spectral-spatial features considering their spatial locations, their spectral signatures and also their possible transformations in a more efficiently way in comparison with SSRN. Precisely, this is the fact that enhances the generalization ability of the network, because the corresponding spectral-spatial features are complemented with important information about characteristic data transformations as a set of instantiation parameters, which eventually allows characterizing the HSI data at a higher abstraction level.

Additionally, Tables VII-VIII give an experimental comparison among the 3D-CNN [25], DFCNN [30] and the proposed approach, using the IP and UP datasets and considering multiple input spatial sizes. In particular, the first column shows the class labels, the second row indicates the number of training samples, and the last three rows provide the OA results for 3D-CNN, DFCNN and the proposed approach, respectively, with different spatial sizes.

Some important observations can be made from Tables VII-VIII. In general, in these tables it is possible to see that larger spatial sizes for the input patches generally result in higher accuracy values (the larger the input size, the more spatial information is considered to complement the spectral data). However, it can be also observed that the proposed approach requires substantially smaller input patches to generate similar or even better accuracy results than the other methods. Precisely, this point reinforces the aforementioned observations concerning the higher generalization capability of the proposed approach. In the case of the IP dataset, 3D-CNN and DFCNN obtain an OA of 97.56 and 97.87 using 27×27 and 29×29 input spatial patches, respectively. In turn, the proposed network is able to achieve a remarkable performance improvement, reaching a 98.69 value, using only a 9×9 input spatial patch. A similar trend can be also observed in the experiments with the the UP dataset. This suggests that the proposed approach is able to uncover more descriptive features than the 3D-CNN and DFCNN techniques.

For illustrative purposes, Fig. 9 shows the classification maps obtained by the DFCNN [30] and the proposed approach for the UP dataset. A visual comparison of both maps indicates that the proposed method provides better class delineation and definition of urban features. Specifically, class boundaries are noticeably more precise and defined. This is particularly the case for classes representing typically urban features, such as self-blocking bricks (in blue), which appears better delineated in the classification map provided by the proposed approach. In addition, the bitumen class (in dark green) contains circular and rectangular urban features that appear better delineated in the map produced by the proposed approach than in the one produced by the DFCNN. Also, the classification results obtained by the proposed approach over unlabeled image areas appears more visually consistent and with better delineated features, which also suggests the higher generalization ability of the proposed network.

3) Experiment 3: In a final experiment we evaluate the convergence of the proposed network architecture. In this context, it is important to note that the proposed network architecture makes use of several innovative building blocks that are able to estimate the probability that a specific spectral-spatial feature occurs in the input HSI data and also its corresponding instantiation parameters, that is, the potential transformations suffered by the corresponding constituent feature on the observable input data. As a result, the HSI features can be intrinsically managed at a higher abstraction level throughout the network because traditional convolutional features are decomposed into canonical spectral-spatial features and their possible transformations, which eventually leads to a significant reduction of the architecture complexity and, therefore, to a good model convergence. To illustrate this point, Fig. 10 displays the evolution of the proposed approach test accuracy per epoch (left side) and computational time in seconds (right side). As it can be seen in Fig. 10, the proposed network only requires a reduced number of epochs and a very short time to reach almost optimal performance, which highlights the remarkably fast convergence of the proposed architecture.

In summary, the experiments reported in this section suggest that the proposed approach provides quantitative and qualitative advantages over traditional HSI classifiers (see Tables II-IV and Figs. 6-8) and also over some of the most relevant state-of-the-art spectral-spatial classification techniques, i.e. 3D-CNN [25], SSRN [35] and DFCNN [30] (see Tables VI-VIII and Figs. 9-10). The proposed method is able to achieve the best global performance in all the considered experimental scenarios, exhibiting relevant performance improvements when considering reduced input patch spatial sizes. The proposed approach seems to provide the most robust behavior with different input patch spatial sizes, which suggests that it is able to generalize more discriminative features to effectively classify HSI data. Unlike other established deep learning models such as 3D-CNN, SSRN and DFCNN, the constituent units of the proposed architecture (capsules) are designed to uncover canonical spectral-spatial features and their corresponding instantiation parameters, which allow characterizing the HSI data at a higher abstraction level while reducing the over-fitting phenomenon inherent to complex and deep networks.

V. Conclusions and Future Lines

In this paper, a new deep learning architecture based on the concept of capsules is presented to effectively classify remotely sensed HSI data. Specifically, the proposed network is composed by a set of spectral-spatial capsule units which characterize the input data at a higher abstraction level by expressing the HSI features as a collection of canonical spectral-spatial patterns and their corresponding instantiation parameters. In this way, the features uncovered by the network become more informative, which eventually leads to a reduction of the architecture complexity and, therefore, to a more accurate model convergence. The experimental comparisons conducted in this work, which consider five well-known HSI datasets and extcolorblackeight established methods, reveal that the proposed approach exhibits competitive advantages with respect to state-of-the-art classification methods.

An important characteristic of the proposed approach is its potential to deal with the inherent complexity of HSI datasets generated by their high spectral resolution. In general, experimental results have shown that the proposed model is able to extract a more relevant and complete information about HSI data cubes by managing spectral-spatial features at a higher abstraction level. Specifically, the spectral-spatial capsule units model the different transformations present in the HSI domain by means of a neuron hierarchy which disentangle the spectral-spatial canonical features from the data transformation parameters. Therefore, the activation of higher-level spectral-spatial features can be conducted by agreement between lower-level features in order to intrinsically model complex connections to better characterize the HSI data, obtaining consistently high classification performance with a limited amount of training data.

Acknowledgment

This work has been supported by Ministerio de Educación (Resolución de 26 de diciembre de 2014 y de 19 de noviembre de 2015, de la Secretaría de Estado de Educación, Formación Profesional y Universidades, por la que se convocan ayudas para la formación de profesorado universitario, de los subprogramas de Formación y de Movilidad incluidos en el Programa Estatal de Promoción del Talento y su Empleabilidad, en el marco del Plan Estatal de Investigación Científica y Técnica y de Innovación 2013-2016. This work has also been supported by Junta de Extremadura (decreto 297/2014, ayudas para la realización de actividades de investigación y desarrollo tecnológico, de divulgación y de transferencia de conocimiento por los Grupos de Investigación de Extremadura, Ref. GR15005). This work has been additionally supported by the Generalitat Valenciana through the contract APOSTD/2017/007 and by the Spanish Ministry of Economy under the projects ESP2016-79503-C2-2-P and TIN2015-63646-C5-5-R.

References

[1] D. Landgrebe, “Hyperspectral image data analysis,” *IEEE Signal processing magazine*, vol. 19, no. 1, pp. 17–28, 2002.

[2] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. Nasrabadi, and J. Chanussot, “Hyperspectral remote sensing data analysis and future challenges,” *IEEE Geoscience and remote sensing magazine*, vol. 1, no. 2, pp. 6–36, 2013.

[3] G. Camps-Valls, D. Tuia, L. Bruzzone, and J. A. Benediktsson, “Advances in hyperspectral image classification: Earth monitoring with statistical learning methods,” *IEEE signal processing magazine*, vol. 31, no. 1, pp. 45–54, 2014.

[4] A. Ghiyamat and H. Z. Shafri, “A review on hyperspectral remote sensing for homogeneous and heterogeneous forest biodiversity assessment,” *International Journal of Remote Sensing*, vol. 31, no. 7, pp. 1837–1856, 2010.

[5] M. Fauvel, Y. Tarabalka, J. A. Benediktsson, J. Chanussot, and J. C. Tilton, "Advances in spectral-spatial classification of hyperspectral images," *Proceedings of the IEEE*, vol. 101, no. 3, pp. 652–675, 2013.

[6] X. Zhang, Y. Sun, K. Shang, L. Zhang, and S. Wang, "Crop classification based on feature band set construction and object-oriented approach using hyperspectral images," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 9, no. 9, pp. 4117–4128, 2016.

[7] K. Manjunath, S. Ray, and D. Vyas, "Identification of indices for accurate estimation of anthocyanin and carotenoids in different species of flowers using hyperspectral data," *Remote Sensing Letters*, vol. 7, no. 10, pp. 1004–1013, 2016.

[8] B. Uzkent, A. Rangnekar, and M. J. Hoffman, "Aerial vehicle tracking by adaptive fusion of hyperspectral likelihood maps," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*. IEEE, 2017, pp. 233–242.

[9] E. G. Njoku, *Encyclopedia of Remote Sensing*. Springer, 2014.

[10] P. Ghamisi, J. Plaza, Y. Chen, J. Li, and A. J. Plaza, "Advanced spectral classifiers for hyperspectral images: A review," *IEEE Geoscience and Remote Sensing Magazine*, vol. 5, no. 1, pp. 8–32, 2017.

[11] G. Camps-Valls and L. Bruzzone, "Kernel-based methods for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 6, pp. 1351–1362, 2005.

[12] J. Haut, M. Paoletti, J. Plaza, and A. Plaza, "Cloud implementation of the K-means algorithm for hyperspectral image analysis," *Journal of Supercomputing*, vol. 73, no. 1, 2017.

[13] Y. Bazi and F. Melgani, "Gaussian process approach to remote sensing image classification," *IEEE transactions on geoscience and remote sensing*, vol. 48, no. 1, pp. 186–197, 2010.

[14] J. Ham, Y. Chen, M. M. Crawford, and J. Ghosh, "Investigation of the random forest framework for classification of hyperspectral data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 3, pp. 492–501, 2005.

[15] J. M. Haut, M. E. Paoletti, J. Plaza, and A. Plaza, "Fast dimensionality reduction and classification of hyperspectral images with extreme learning machines," *Journal of Real-Time Image Processing*, Jun 2018. [Online]. Available: https://doi.org/10.1007/s11554-018-0793-9

[16] Y. Chen, Z. Lin, X. Zhao, G. Wang, and Y. Gu, "Deep learning-based classification of hyperspectral data," *IEEE Journal of Selected topics in applied earth observations and remote sensing*, vol. 7, no. 6, pp. 2094–2107, 2014.

[17] D. L. Donoho *et al.*, "High-dimensional data analysis: The curses and blessings of dimensionality," *AMS Math Challenges Lecture*, vol. 1, p. 32, 2000.

[18] Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning," *Nature*, vol. 521, p. 436444, May 2015.

[19] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[20] 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.

[21] C. Zhao, X. Wan, G. Zhao, B. Cui, W. Liu, and B. Qi, "Spectral-spatial classification of hyperspectral imagery based on stacked sparse autoencoder and random forest," *European Journal of Remote Sensing*, vol. 50, no. 1, pp. 47–63, 2017.

[22] T. Li, J. Zhang, and Y. Zhang, "Classification of hyperspectral image based on deep belief networks," in *IEEE Int. Conf. Image Proces.,*, 2014, pp. 5132–5136.

[23] X. Chen, S. Xiang, C.-L. Liu, and C.-H. Pan, "Vehicle Detection in Satellite Images by Hybrid Deep Convolutional Neural Networks," *IEEE Geosci. Remote Sens. Lett.*, vol. 11, no. 10, pp. 1797–1801, Oct. 2014.

[24] X. Ma, H. Wang, and J. Geng, "Spectral-Spatial Classification of Hyperspectral Image Based on Deep Auto-Encoder," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 9, no. 9, pp. 4073–4085, Feb. 2016.

[25] Y. Chen, H. Jiang, C. Li, X. Jia, and P. Ghamisi, "Deep Feature Extraction and Classification of Hyperspectral Images Based on Convolutional Neural Networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 10, pp. 6232–6251, Oct. 2016.

[26] W. Li, G. Wu, F. Zhang, and Q. Du, "Hyperspectral image classification using deep pixel-pair features," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 2, pp. 844–853, Feb. 2017.

[27] W. Zhao and S. Du, "Spectral-Spatial Feature Extraction for Hyperspectral Image Classification: A Dimension Reduction and Deep Learning Approach," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 8, pp. 4544–4554, Aug. 2016.

[28] J. Yang, Y. Zhao, J. C. W. Chan, and C. Yi, "Hyperspectral image classification using two-channel deep convolutional neural network," in *IEEE Int. Geosci. Remote Sens. Symp.*, 2016, pp. 5079–5082.

[29] H. Zhang, Y. Li, Y. Zhang, and Q. Shen, "Spectral-spatial classification of hyperspectral imagery using a dual-channel convolutional neural network," *Remote Sens. Lett.*, vol. 8, no. 5, pp. 438–447, Jan. 2017.

[30] M. E. Paoletti, J. M. Haut, J. Plaza, and A. Plaza, "A new deep convolutional neural network for fast hyperspectral image classification," *ISPRS Journal of Photogrammetry and Remote Sensing*, 2017.

[31] J. Acquarelli, E. Marchiori, L. M. C. Buydens, T. N. Tran, and T. van Laarhoven, "Convolutional neural networks and data augmentation for spectral-spatial classification of hyperspectral images," *CoRR*, vol. abs/1711.05512, 2017. [Online]. Available: http://arxiv.org/abs/1711.05512

[32] D. E. Worrall, S. J. Garbin, D. Turmukhambetov, and G. J. Brostow, “Harmonic Networks: Deep Translation and Rotation Equivariance,” 2016.

[33] R. K. Srivastava, K. Greff, and J. Schmidhuber, “Training Very Deep Networks,” *CoRR*, vol. abs/1507.06228, 2015.

[34] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[35] Z. Zhong, J. Li, Z. Luo, and M. Chapman, “Spectral-spatial residual network for hyperspectral image classification: A 3-d deep learning framework,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 2, pp. 847–858, Feb 2018.

[36] G. Huang, Z. Liu, and K. Q. Weinberger, “Densely connected convolutional networks,” *CoRR*, vol. abs/1608.06993, 2016. [Online]. Available: http://arxiv.org/abs/1608.06993

[37] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, “Aggregated residual transformations for deep neural networks,” in *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE, 2017, pp. 5987–5995.

[38] S. Sabour, N. Frosst, and G. E. Hinton, “Dynamic routing between capsules,” in *Advances in Neural Information Processing Systems*, 2017, pp. 3859–3869.

[39] G. E. Hinton, A. Krizhevsky, and S. D. Wang, “Transforming auto-encoders,” in *Artificial Neural Networks and Machine Learning – ICANN 2011*, T. Honkela, W. Duch, M. Girolami, and S. Kaski, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 44–51.

[40] P. Fisher, “The pixel: a snare and a delusion,” *International Journal of Remote Sensing*, vol. 18, no. 3, pp. 679–685, 1997.

[41] J. M. Haut, M. E. Paoletti, J. Plaza, J. Li, and A. Plaza, “Active learning with convolutional neural networks for hyperspectral image classification using a new bayesian approach,” *IEEE Transactions on Geoscience and Remote Sensing*, 2018.

[42] D. J. Field, “Wavelets, vision and the statistics of natural scenes,” *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 357, no. 1760, pp. 2527–2542, 1999.

[43] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, “What is the best multi-stage architecture for object recognition?” in *2009 IEEE 12th International Conference on Computer Vision*, Sept 2009, pp. 2146–2153.

[44] V. Nair and G. E. Hinton, “Rectified Linear Units Improve Restricted Boltzmann Machines,” in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, Johannes Fürnkranz and Thorsten Joachims, Ed. Omnipress, 2010, pp. 807–814.

[45] X. Glorot, A. Bordes, and Y. Bengio, “Deep Sparse Rectifier Neural Networks,” in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS-11)*, Geoffrey J. Gordon and David B. Dunson, Ed. Journal of Machine Learning Research - Workshop and Conference Proceedings, 2011, pp. 315–323.

[46] D. Yu, H. Wang, P. Chen, and Z. Wei, “Mixed pooling for convolutional neural networks,” in *Rough Sets and Knowledge Technology*, D. Miao, W. Pedrycz, D. Ślzak, G. Peters, Q. Hu, and R. Wang, Eds. Cham: Springer International Publishing, 2014, pp. 364–375.

[47] M. D. Zeiler and R. Fergus, “Stochastic pooling for regularization of deep convolutional neural networks,” *CoRR*, vol. abs/1301.3557, 2013. [Online]. Available: http://arxiv.org/abs/1301.3557

[48] T. Williams and R. Li, “Wavelet pooling for convolutional neural networks,” in *International Conference on Learning Representations*, 2018.

[49] Y. T. Zhou and R. Chellappa, “Computation of optical flow using a neural network,” in *IEEE 1988 International Conference on Neural Networks*, July 1988, pp. 71–78 vol.2.

[50] G. E. Hinton, S. Sabour, and N. Frosst, “Matrix capsules with EM routing,” in *International Conference on Learning Representations*, 2018.

[51] D. P. Kingma and J. L. Ba, “ADAM: A method for stochastic optimization,” *CoRR*, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

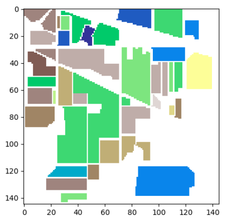
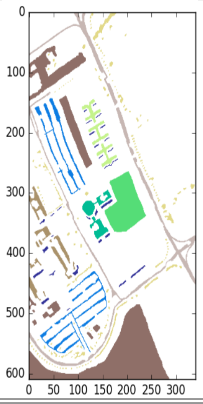
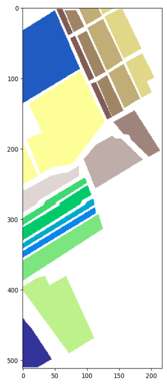
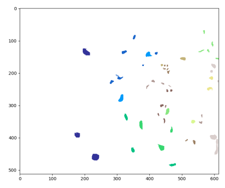
[52] R. O. Green, M. L. Eastwood, C. M. Sarture, T. G. Chrien, M. Aronsson, B. J. Chippendale, J. A. Faust, B. E. Pavri, C. J. Chovit, M. Solis, M. R. Olah, and O. Williams, “Imaging spectroscopy and the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS),” *Remote Sensing of Environment*, vol. 65, no. 3, pp. 227–248, 1998. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0034425798000649

[53] B. Kunkel, F. Blechinger, R. Lutz, R. Doerffer, H. van der Piepen, and M. Schroder, “ROSIS (Reflective Optics System Imaging Spectrometer) - A candidate instrument for polar platform missions,” in *Proc. SPIE 0868 Optoelectronic technologies for remote sensing from space*, J. Seeley and S. Bowyer, Eds., 1988, p. 8.

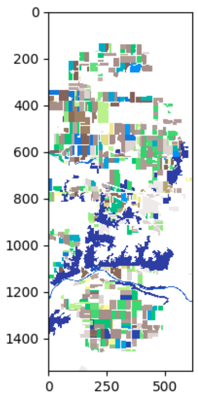
[54] B. Waske, S. van der Linden, J. A. Benediktsson, A. Rabe, and P. Hostert, “Sensitivity of support vector machines to random feature selection in classification of hyperspectral data,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, no. 7, pp. 2880–2889, 2010.

[55] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, “Automatic differentiation in pytorch,” 2017.

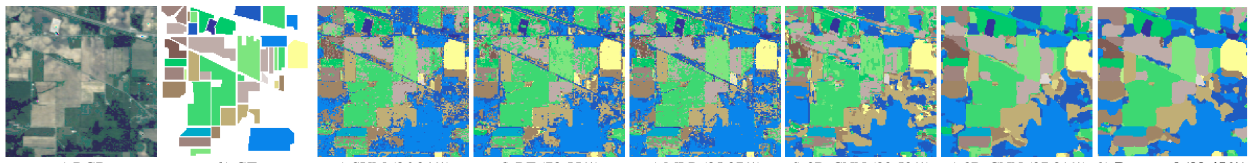
Fig. 5. Number of Available Samples in the Indian Pines (IP), University of Pavia (UP) and Salinas Valley (SV) HSI datasets.

INDIAN PINES (IP)			UNIVERSITY OF PAVIA (UP)			SALINAS VALLEY (SV)			KENNEDY SPACE CENTER (KSC)		
											
Color	Land-cover type	Samples	Color	Land-cover type	Samples	Color	Land-cover type	Samples	Color	Land-cover type	Samples
	Background	10776		Background	164624		Background	56975		Background	309157
⬜	Alfalfa	46	🟫	Asphalt	6631	⬜	Brocoli-green-weeds-1	2009	⬜	Scrub	761
🟫	Corn-notill	1428	🟫	Meadows	18649	🟫	Brocoli-green-weeds-2	3726	🟫	Willow-swamp	243
🟫	Corn-min	830	🟫	Gravel	2099	🟫	Fallow	1976	🟫	CP-hammock	256
🟫	Corn	237	🟨	Trees	3064	🟫	Fallow-rough-plow	1394	🟫	Slash-pine	252
🟫	Grass/Pasture	483	🟩	Painted metal sheets	1345	🟫	Fallow-smooth	2678	🟫	Oak/Broadleaf	161
🟫	Grass/Trees	730	🟩	Bare Soil	5029	🟫	Stubble	3959	🟨	Hardwood	229
🟨	Grass/pasture-mowed	28	🟩	Bitumen	1330	🟨	Celery	3579	🟩	Swap	105
🟨	Hay-windrowed	478	🟦	Self-Blocking Bricks	3682	🟨	Grapes-untrained	11271	🟩	Graminoid-marsh	431
🟩	Oats	20	🟪	Shadows	947	🟩	Soil-vinyard-develop	6203	🟩	Spartina-marsh	520
🟩	Soybeans-notill	972				🟩	Corn-senesced-green-weeds	3278	🟩	Cattail-marsh	404
🟩	Soybeans-min	2455				🟩	Lettuce-romaine-4wk	1068	🟦	Salt-marsh	419
🟩	Soybean-clean	593				🟩	Lettuce-romaine-5wk	1927	🟦	Mud-flats	503
🟦	Wheat	205				🟦	Lettuce-romaine-6wk	916	🟪	Water	927
🟦	Woods	1265				🟦	Lettuce-romaine-7wk	1070			
🟦	Bldg-Grass-Tree-Drives	386				🟦	Vinyard-untrained	7268			
🟪	Stone-steel towers	93				🟪	Vinyard-vertical-trellis	1807			
	Total samples	21025		Total samples	207400		Total samples	111104		Total samples	314368

BIG INDIAN PINES SCENE (BIP)

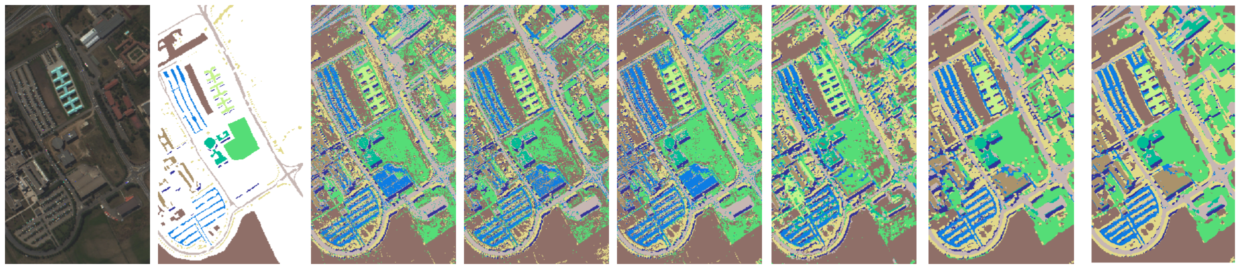


Color	Land cover type	Samples	Color	Land cover type	Samples
	Background	1310047	⬜	BareSoil	57
⬜	Buildings	17195	⬜	Concrete/Asphalt	69
⬜	Corn	17783	⬜	Corn?	158
🟫	Corn-EW	514	🟫	Corn-NS	2356
🟫	Corn-CleanTill	12404	🟫	Corn-CleanTill-EW	26486
🟫	Corn-CleanTill-NS	39678	🟫	Corn-CleanTill-NS-Irrigated	800
🟫	Corn-CleanTilled-NS?	1728	🟫	Corn-MinTill	1049
🟫	Corn-MinTill-EW	5629	🟫	Corn-MinTill-NS	8862
🟫	Corn-NoTill	4381	🟫	Corn-NoTill-EW	1206
🟫	Corn-NoTill-NS	5685	🟫	Fescue	114
🟫	Grass	1147	🟫	Grass/Trees	2331
🟫	Grass/Pasture-mowed	19	🟫	Grass/Pasture	73
🟨	Grass-runway	37	🟨	Hay	1128
🟨	Hay?	2185	🟨	Hay-Alfalfa	2258
🟨	Lake	224	🟨	NotCropped	1940
🟨	Oats	1742	🟨	Oats?	335
🟩	Orchard	39	🟩	Pasture	10386
🟩	pond	102	🟩	Soybeans	9391
🟩	Soybeans?	894	🟩	Soybeans-NS	1110
🟩	Soybeans-CleanTill	5074	🟩	Soybeans-CleanTill?	2726
🟩	Soybeans-CleanTill-EW	11802	🟩	Soybeans-CleanTill-NS	10387
🟩	Soybeans-CleanTill-Drilled	2242	🟩	Soybeans-CleanTill-Weedy	543
🟩	Soybeans-Drilled	15118	🟩	Soybeans-MinTill	2667
🟦	Soybeans-MinTill-EW	1832	🟦	Soybeans-MinTill-Drilled	8098
🟦	Soybeans-MinTill-NS	4953	🟦	Soybeans-NoTill	2157
🟦	Soybeans-NoTill-EW	2533	🟦	Soybeans-NoTill-NS	929
🟦	Soybeans-NoTill-Drilled	8731	🟦	Swampy Area	583
🟦	River	3110	🟦	Trees?	580
🟪	Wheat	4979	🟪	Woods	63562
🟪	Woods?	144			
	Total samples 1644292				



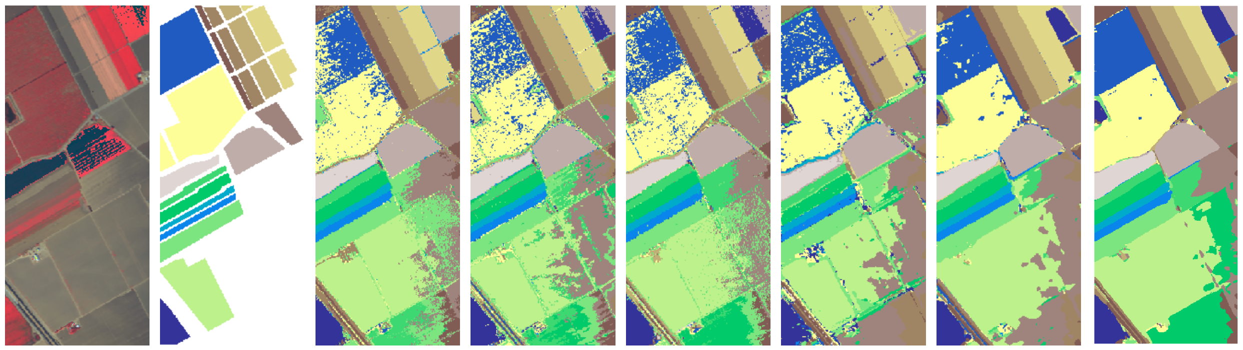
a) RGB *b*) GT *c*) SVM (86.24%) *d*) RF (78.55%) *e*) MLP (85.27%) *f*) 2D-CNN (83.59%) *g*) 3D-CNN (97.81%) *h*) **Proposed (99.45%)**

Fig. 6. Classification maps for the Indian Pines (IP) dataset. The first image (a) represents a simulated RGB composition of the scene. The second one (b) contains the ground-truth classification map. Finally, images from (c) to (h) provide the classification maps corresponding to Table II. Note that the overall classification accuracies are shown in brackets and the best result is highlighted in bold font.



a) RGB *b*) GT *c*) SVM (95.20%) *d*) RF (92.03%) *e*) MLP (94.82%) *f*) 2D-CNN (94.77%) *g*) 3D-CNN (98.54%) *h*) **Proposed (99.95%)**

Fig. 7. Classification maps for the University of Pavia (UP) dataset. The first image (a) represents a simulated RGB composition of the scene. The second one (b) contains the ground-truth classification map. Finally, images from (c) to (h) provide the classification maps corresponding to Table III. Note that the overall classification accuracies are shown in brackets and the best result is highlighted in bold font.



a) RGB *b*) GT *c*) SVM (94.15%) *d*) RF (90.76%) *e*) MLP (93.87%) *f*) 2D-CNN (92.31%) *g*) 3D-CNN (97.44%) *h*) **Proposed (99.81%)**

Fig. 8. Classification maps for the Salinas Valley (SV) dataset. The first image (a) represents a simulated RGB composition of the scene. The second one (b) contains the ground-truth classification map. Finally, images from (c) to (h) provide the classification maps corresponding to Table IV. Note that the overall classification accuracies are shown in brackets and the best result is highlighted in bold font.

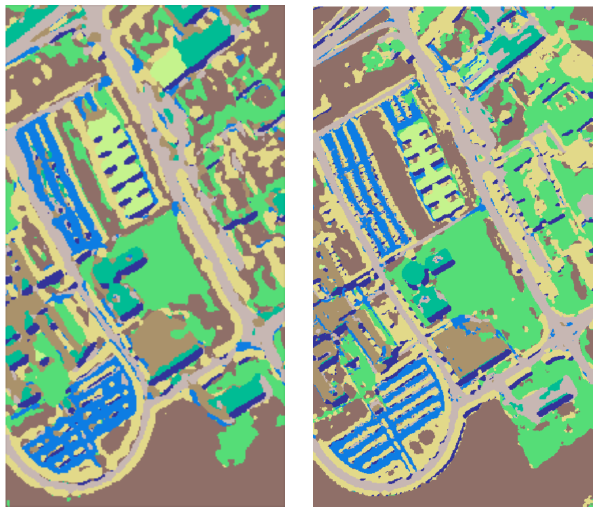


Fig. 9. Classification maps obtained by the DFCNN [30] (left) and the proposed approach (right) for the University of Pavia dataset. A visual comparison of both maps indicates that the proposed method provides better class delineation and definition of urban features, for instance in classes such as self-blocking bricks (blue) or bitumen (dark green), containing both circular and rectangular urban features.

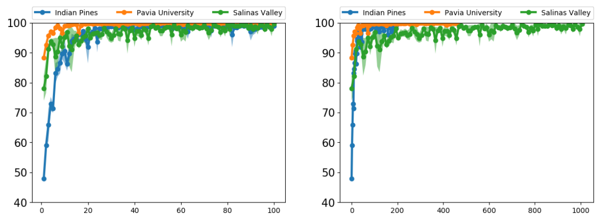


Fig. 10. Evolution of the test accuracy (in %) of the proposed approach (y-axis) versus epochs (left) and computational time in seconds (right) for the experiments with the IP, UP and SV datasets.