FLOP-Reduction Through Memory Allocations Within CNN for Hyperspectral Image Classification

Mercedes E. Paoletti, *Member, IEEE*, Juan M. Haut, *Senior Member, IEEE*, Xuanwen Tao, *Student Member, IEEE*, Javier Plaza, *Senior Member, IEEE*, and Antonio Plaza, *Fellow, IEEE*

***Abstract*— Convolutional neural networks (CNNs) have proven to be a powerful tool for the classification of hyperspectral images (HSIs). The CNN kernels are able to naturally include spatial information to smooth out the spectral variability and the noise present in HSI data. However, these kernels are composed of a large number of learning parameters that must be correctly adjusted to achieve good performance. This forces the model to consume a large amount of training data, being prone to overfitting when limited labeled samples are available. In addition, the execution of kernels is computationally very expensive, increasing quadratically with respect to the size of the convolution filter. This significantly reduces the performance of the model. To overcome the aforementioned limitations, this work presents a new few-parameter CNN (based on shift operations) for HSI classification that dramatically reduces both the number of parameters and the computational complexity of the model in terms of floating-point operations (FLOPs). The operational module combines a shift kernel (which adjusts the input data in particular directions without involving any parameters nor FLOPs) with pointwise convolutions that perform the feature extraction stage. The newly developed shift-based CNN has been employed to conduct HSI classification over five widely used and challenging data sets, achieving very promising results in terms of computational performance and classification accuracy.**

***Index Terms*— Classification, convolutional neural networks (CNNs), hyperspectral images (HSIs), shift operation.**

Manuscript received March 29, 2020; revised August 6, 2020; accepted September 12, 2020. This work was supported in part by the Ministerio de Educación (Resolución 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), in part by the Junta de Extremadura (Decreto 14/2018, de 6 de febrero, por el que se establecen las bases reguladoras de las 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. GR18060), in part by the Acción III (University of Extremadura), and in part by the European Union's Horizon 2020 research and innovation program under Grant 734541 (EOXPOSURE). *(Corresponding author: Mercedes E. Paoletti.)*

The authors are with Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, Escuela Politécnica, University of Extremadura, 10003 Cáceres, Spain (e-mail: mpaoletti@unex.es; juanmariohaut@unex.es; taoxuanwenupc@gmail.com; jplaza@unex.es; aplaza@unex.es).

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

Digital Object Identifier 10.1109/TGRS.2020.3024730

I. Introduction

ADVANCES in computing technology (both in terms of data processing and storage) have led to a genuine revolution in the field of artificial intelligence, allowing the development and implementation of complex and really sophisticated automatic data processing methods. Machine learning approaches have reached a high level of specialization in data analysis and pattern recognition [1], being successfully applied to a wide range of activities, including speech recognition [2]–[4], text analysis [5], [6], data mining [7], [8] or image processing [9], [10], and computer vision [11], [12], among others. In the field of Earth observation (EO) and remote sensing, multiple algorithms have been applied for the analysis of remotely sensed data [13]–[15], achieving accurate results in tasks, such as object detection [16], [17] and land-cover classification [18].

A. HSI Classification: Challenges

In remote sensing, the data captured by imaging spectrometers are particularly interesting due to a large amount of spectral–spatial information comprised in their data products. In fact, the potential of hyperspectral images (HSIs) lies in the ability to simultaneously capture hundreds of images from the same area on the Earth's surface, by measuring the reflectance of terrestrial materials at different wavelength channels along the electromagnetic spectrum [19], usually covering the visible, near-infrared (NIR), and shortwave infrared (SWIR) spectrum [20] between 400 and 2500 nm. As a result, the HSI scene forms a multidimensional data cube, where HSI pixels contain the contiguous reflectance spectra of the observed materials. These spectral signatures can be understood as fingerprints, being each one unique associated with each type of material, and allowing for a detailed characterization of the surface of the Earth. The recent literature contains a large number of scientific works that take advantage of the great amount of information contained in HSI data cubes in tasks related to pattern recognition [21], [22]. In particular, HSI data have been widely used for land-cover classification tasks, where well-known machine learning techniques have been traditionally adapted and applied to categorize the content of each HSI pixel. In this sense, popular classifiers, such as support vector

machines (SVMs) [23], random forests (RFs) [24], artificial neural networks (ANNs) [25], or multinomial logistic regression (MLR) [26], have been usually implemented as pixelwise methods to exploit the rich spectral information contained by each HSI sample in an isolated way. However, the performance of these methods suffers from several limitations due to the intrinsic characteristics of HSI data (e.g., the curse of dimensionality, noise, and spectral variability).

On the one hand, the spatial resolution of HSI scenes is usually low compared with that of other EO instruments, so the spectral composition of a pixel is, in fact, a combination of the signatures of several materials that form that pixel. Also, variations in surface illumination introduce changes in spectral signatures, which can be attenuated depending on direct lighting (sunny areas) or reflected/scattered radiation (shaded areas). In addition, significant distortions are introduced into the spectra during the acquisition process due to sensor inaccuracies and uncontrolled environmental disturbances. As a result, HSI data exhibits high intraclass variability and interclass similarity, hampering the reliability of spectral classifiers.

On the other hand, the large spectral dimensionality can also further complicate the classification process, as it imposes significant storage and processing restrictions [27], [28] while increasing exponentially the volume of the feature space, which, in turn, makes available data widely scattered (the so-called peaking paradox [29]–[31]). This leads to two major implications: first, more training data are needed to reliably cover all spectral features; second, more parameters has to be introduced in the classification method. By adding more parameters to the classifier, the error estimation becomes also more complex, hampering the optimization of the loss between the desired classification result and the one obtained by the model, i.e., the available samples are not enough to accurately estimate the statistical parameters that define the land-cover classes present into the scene. As a result, spectral-based classification methods quickly over/under-fit their performance [32]. This, coupled with the high variability of the spectral signatures, turns the HSI data classification task into an ill-posed problem that is highly affected by the curse of dimensionality [23], [33]. Moreover, the increase in the number of statistical parameters (together with the high dimensionality of the HSI data) results in high memory consumption, increasing also the number of arithmetic operations employed by the model, so the computational load is negatively affected.

B. CNNs for HSI Data Classification

In this context, algorithms inspired by deep learning [34] are able to handle such dimensionality issues in a more effective way through the hierarchical learning of deep features extracted from the data [35], yielding a wide range of models with great generalization and expressivity properties [36], [37] that provide state-of-the-art predictive capabilities in many research fields [38], [39]. In particular, due to computing advances in both hardware platforms and software frameworks, deep learning classifiers have profoundly impacted the remotely sensed HSI classification field in recent years [40], [41], being the convolutional neural network (CNN) a highly representative classifier [42]–[48] due to its ability to extract and learn deep and abstract feature representations of the original input data, being able to model complex nonlinear relationship within the data. Its n-dimensional kernel-based architecture (with $n = 1$, 2, or 3 depending on whether it is applied to the spectral, spatial, or spectral-space dimensions) allows not only for the exploitation of the spectral content of the HSI scene but also for the natural integration of spectral and spatial information, which has proven to reduce the classification uncertainty by combining each pixel spectra with the spatial-contextual information provided by its neighbors (such as object shapes, textures, and geometrical structures) [49], [50]. As a result, convolutional-based models for HSI data classification are now able to achieve excellent performance, positioning themselves as the current state of the art in the field [51]–[53].

C. Computational Complexity of CNNs in HSI Classification and Existing Optimizations

Despite the outstanding results obtained by CNN-inspired architectures, these models rely on 2-D/3-D convolution layers, where the lth layer defines a kernel as the data tensor $\mathbf{W}^{(l)} \in \mathbb{R}^{n_k^{(l)} \times n_k^{(l)} \times n_f^{(l-1)} \times n_f^{(l)}}$ composed of $n_f^{(l)}$ filters with spatial size $n_k^{(l)} \times n_k^{(l)}$ and channel size $n_f^{(l-1)}$ that overlap and slide the input volume through a stride parameter, as a sliding-window algorithm, with the purpose of aggregating the spatial and spectral information contained into the HSI scene. Some of the most usual spatial kernel sizes range from 3×3 to 11×11 and 29×29 [54]. In this regard, it should be noted that each convolution layer involves $((n_k^{(l)} \cdot n_k^{(l)}) + 1) \cdot n_f^{(l)}$ parameters at least, which, in a deep architecture, means that millions of parameters must be not only correctly adjusted but also computed along with the feature volumes. In fact, each CNN model exhibits the complexity indicated by the following equation:

$$\mathrm{O}\left(\sum_{l=1}^{L} n_f^{(l-1)} \cdot n_k^{(l)} \cdot n_k^{(l)} \cdot n_f^{(l)} \cdot m^{(l)} \cdot m^{(l)}\right) \tag{1}$$

where l corresponds to the index of the current layer, L being the number of convolution layers (i.e., the depth of the model), $n_f^{(l-1)}$ and $n_f^{(l)}$ are the number of filters of the $(l-1)$th and lth layers, respectively (i.e., the width of the layers),[1] $n_k^{(l)}$ is the spatial size of the current layer (i.e., the length), and $m^{(l)}$ is the spatial size of the resulting output feature volume [55]. In this sense, it is easy to observe that spatial convolutions $n_k \times n_k$ are quite expensive, increasing both the computational time and the model's size quadratically with respect to n_k [56]. This can be significantly exacerbated by the large dimensionality of HSI data, involving an expensive computational burden.

Although some strategies have been developed to reduce the spatial size of the convolution layers through bottleneck-based techniques [57], distributing the computing load between

[1]It must be noted that, as $n_f^{(l-1)}$ is the number of filters of the previous layer, it will indicate also the number of channels contained into the input feature volume of the lth layer.

all layers by controlling the spectral–spatial dimensions of the feature volumes [58], adapting depthwise convolutional implementations [59], [60], or even redesigning the operational blocks as a continuous-time evolving model [61], few efforts have been devoted to reducing the number of model parameters and the floating-point operations (FLOPs)[2] in those deep convolutional-based architectures for HSI data classification.

D. Proposed Contribution and Organization of this Article

To address the aforementioned issues, this article proposes a new efficient CNN model for HSI data classification, which is inspired by parameter-free and zero-FLOP convolution layers [56]. In particular, instead of sliding kernels, the operating layers developed to perform spectral–spatial feature extraction are based on shift operations with pointwise convolutions [56], which replicates the movement through each feature volume's channel across different spatial directions while performing the aggregation of the spatial information through 1×1 convolutions. As the shift operation requires neither FLOPs nor parameters, the resulting network is more efficient in terms of computational performance and memory consumption. Also, precisely, because the proposed model does not require trainable parameters, the complexity of the network is also reduced, avoiding the degradation of classification results when training with very few labeled samples.

The proposed model has been tested using five popular HSI scenes and compared with the traditional CNN model. The obtained results demonstrate that our newly proposed approach is able to reach similar performance in terms of accuracy while significantly reducing both the number of parameters and the FLOPs employed.

The remainder of this article is organized as follows. Section II introduces the methodology employed by our newly proposed shift-based convolution layers for HSI data classification. Section III validates the performance of the new model by providing a detailed discussion of the results obtained using five different HSI data sets to perform a comparison with the current state-of-the-art HSI classifiers. Section IV concludes this article with some remarks and hints at plausible future works.

II. Proposed Methodology

A. Spatial Convolution

CNN models can be regarded as a deep stack of L operational blocks where, considering an input HSI data volume denoted by $\mathbf{X}^{(l-1)} \in \mathbb{R}^{m^{(l-1)} \times m^{(l-1)} \times n_f^{(l-1)}}$, the lth block performs a feature extraction stage composed of two main steps, i.e., the data transformation of the input volume and the generation of the corresponding neuronal responses as an output feature volume $\mathbf{X}^{(l)} \in \mathbb{R}^{m^{(l)} \times m^{(l)} \times n_f^{(l)}}$ through convolutional and linear/nonlinear activation layers (see the following equation):

$$\mathbf{X}^{(l)} = \mathcal{H}(\mathcal{C}(\mathbf{X}^{(l-1)}, \mathbf{W}^{(l)}, \mathbf{b}^{(l)})) \tag{2}$$

where $\mathcal{H}(\cdot)$ indicates the activation function (usually a rectified linear unit (ReLU) [62]) and $\mathcal{C}(\cdot)$ represents the convolution operation applied over the input data through the current convolutional kernel $\mathbf{W}^{(l)} \in \mathbb{R}^{n_k^{(l)} \times n_k^{(l)} \times n_f^{(l-1)} \times n_f^{(l)}}$ and the bias vector $\mathbf{b}^{(l)} \in \mathbb{R}^{n_f^{(l)}}$. Delving into convolution, this layer can be described as a linear operation that aggregates spatial-contextual information, combining it with spectral characteristics by the sum of dot products between the kernel weights and the input volume data (see Fig. 1). Equation (3) illustrates the computation of the output element (i, j) of the zth filter (being $z = \{1, \ldots, n_f^{(l)}\}$) that belongs to the lth convolution layer

$$x_{i,j,z}^{(l)} = \sum_{\hat{i},\hat{j},\hat{t}} w_{\hat{i},\hat{j},\hat{t},z}^{(l)} \cdot x_{i+\tilde{i},j+\tilde{j},\hat{t}}^{(l-1)} + b_z^{(l)} \tag{3}$$

where i, j and $\hat{i}$, $\hat{j}$ are the spatial indices that cover the input and output volumes and the kernel weights, respectively, being $\tilde{i} = \hat{i} - \lfloor m^{(l-1)}/2 \rfloor$ and $\tilde{j} = \hat{j} - \lfloor m^{(l-1)}/2 \rfloor$ the recentered spatial indices, while z and $\hat{t}$ are the spectral indices that cover the data and weight volumes along the channel dimension. Looking at (2) and (3), we can observe that each convolution layer involves $((n_k^{(l)} \cdot n_k^{(l)}) + 1) \cdot n_f^{(l)}$ parameters, considering the layer's width, length, and the number of biases. In this sense, each layer exhibits a quadratic cost with respect to the kernel length. In addition, the time complexity of the entire CNN model also grows quadratically with respect to the kernel size [see (1)]. This results in a large number of parameters to be stored (ranging from the thousands to the millions) and a huge amount of operations to be computed, imposing many restrictions on both storage and computing resources. To overcome this problem, we introduce a compact shift-and-pointwise strategy similar to the one in [56] and [63] to perform efficient HSI data classification.

B. Avoiding Spatial Convolutions Through the Shift Operation

The proposed shift-based network can be understood as an adaptation of the depthwise convolution [64], [65]. As we can observe in Fig. 1, the depthwise convolution layer divides its operation into two steps: the first one applies a single filter to each input channel, i.e., it slips one filter of size $k^{(l)} \times k^{(l)}$ along a single channel of the input data $\mathbf{X}^{(l-1)}$ to extract spatial features, while the second step applies a pointwise convolution (1×1 convolution) to combine the data along the channels. However, although this two-step strategy can reduce the number of parameters (and also the computational cost), the implementation of depthwise convolution requires a significant amount of memory accesses [66], being the input–output (I/O) memory operations several orders of magnitude slower and more energy-consuming than the convolution's FLOPs. Thus, in the end, the desired optimization of time and resources cannot be achieved.

[2]This article will take as a reference the number of arithmetic operations (also FLOPs) employed by the deep classification methods, to give an approximate estimation of their computational complexity. Note the difference between FLOPs and FLOPS, where the second concept refers to FLOPs per second, which is a measure of a computer's performance, to indicate a rate between the number of operations that the hardware device can perform in one second.

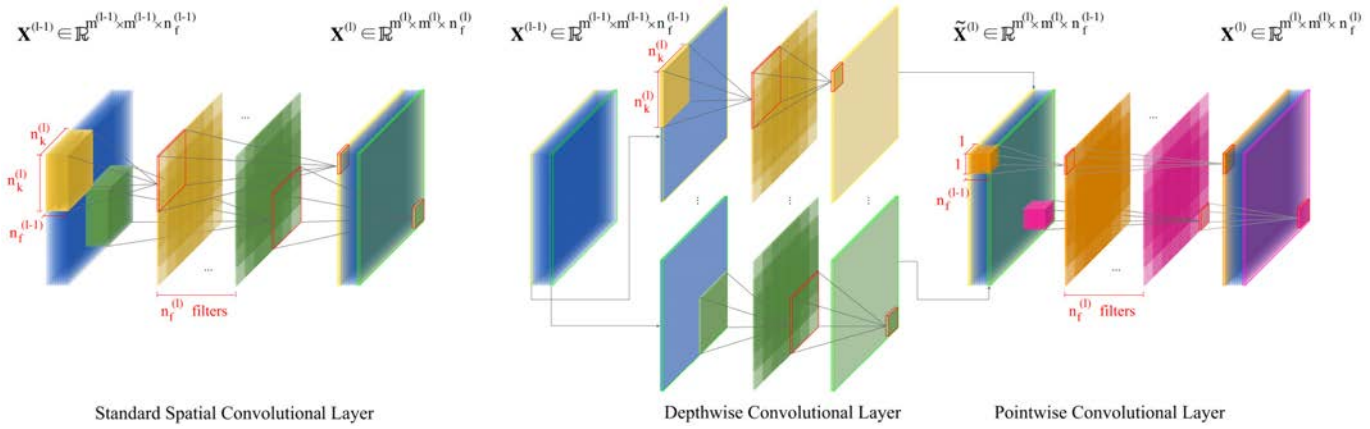


Fig. 1. Illustration of the traditional spatial convolution layer in a CNN architecture—which obtains the output volume $\mathbf{X}^{(l)}$ in a single step by applying a kernel $\mathbf{W}^{(l)} \in \mathbb{R}^{n_k^{(l)} \times n_k^{(l)} \times n_f^{(l-1)} \times n_f^{(l)}}$ —versus a depthwise separable convolution—which obtains the output volume $\mathbf{X}^{(l)}$ in two steps by applying $n_f^{(l-1)}$ separable kernels of size $n_k^{(l)} \times n_k^{(l)} \times 1$—and a pointwise convolution—composed of $n_f^{(l)}$ filters of size $1 \times 1 \times n_f^{(l-1)}$.

To overcome this limitation, the shift-based convolution defines also a two-step process. In the first step, a kernel $\mathbf{W}^{(l)} \in \mathbb{N}^{m \times m \times n_f^{(l-1)}}$ [3] is defined with the same spatial and channel dimensions as those of the input data volume. This kernel is applied by covering the entire $\mathbf{X}^{(l-1)}$ following

$$\tilde{x}_{i,j,t}^{(l)} = \sum_{\hat{i},\hat{j}} w_{\hat{i},\hat{j},t}^{(l)} \cdot x_{i+\tilde{i},j+\tilde{j},t}^{(l-1)} \tag{4}$$

where each $w_{\hat{i},\hat{j},t}^{(l)} \in \{0, 1\}$ depending on

$$w_{\hat{i},\hat{j},t}^{(l)} = \begin{cases} 1, & \text{if } \hat{i} = i_{n_f{}^{(l-1)}} \text{ and } \hat{j} = j_{n_f{}^{(l-1)}} \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Here, $i_{n_f{}^{(l-1)}}$ and $j_{n_f{}^{(l-1)}}$ are two channel-dependent indices that set one of the values of $\mathbf{W}_{:,:,n_f^{(l-1)}}^{(l)}$ to be 1 and the rest to be 0, that is, they are not learnable parameters. Depending on the location of the nonzero element, the shift operation will be performed in one direction or the other. This means that the tth filter in $\mathbf{W}^{(l)}$ (with $t = 1, \ldots, n_f^{(l-1)}$) only contains one nonzero value to indicate the shift direction, so $\mathbf{W}_{:,:,n_f^{(l-1)}}^{(l)}$ can be considered as a shift matrix (see Fig. 2). In this context, the spatial information contained into the input volume is processed, resulting in the output data volume $\tilde{\mathbf{X}} \in \mathbb{R}^{m \times m \times n_f^{(l-1)}}$. Now, we can combine the information across the channel domain by including a pointwise convolution defined by $n_f^{(l)}$ filters of size $1 \times 1 \times n_f^{(l-1)}$ as the second step. In the end, the output data volume $\mathbf{X} \in \mathbb{R}^{m \times m \times n_f^{(l)}}$ is obtained.

Following this, we can avoid the extensive computations required by both standard spatial convolutions and depthwise separable convolutions since the shift operation does not require parameters to be learned and does not involve FLOPs to compute. Instead, it only adjusts the data channels in some directions through some memory operations [56], so it can be considered as a nonarithmetic layer.

[3] In the shift-based convolution layer, both the input and output data volumes can maintain the same spatial dimension; thus, to simplify the notation, we assume $m^{(l-1)} = m^{(l)} = m$, while keeping $f^{(l-1)}$ and $f^{(l)}$ for the channel dimension.

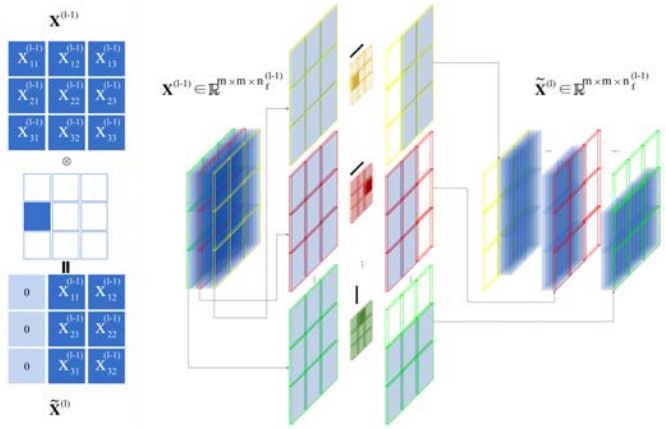


Fig. 2. Illustration of the shift-based convolution layer for an input data volume with spatial size set to $m = 3$. Note that, for a $m \times m$ input volume, m^2 different shift operations (or directions) are allowed. If $m < n_f^{(l-1)}$, all shift directions can be applied to the data. Also, we can group the layers and apply the shift operation over one group following one direction.

C. Implementing the HSI Shift-Based Neural Network

It must be noted that, if we consider $m \times m$ input volumes, m^2 different shift directions are allowed. Moreover, considering input volumes with $n_f^{(l-1)}$ channels, we can obtain $(m^2)^{n_f^{(l-1)}}$ different shift kernels, so the search for the optimal combination of shift kernels can grow quadratically with the spatial size of the data and exponentially with its spectral dimensionality, which makes it prohibitively expensive. To avoid this, grouped-shift is performed [63], where the input channels are divided into $\lfloor n_f^{(l-1)}/m^2 \rfloor$ groups. Then, for all the channels that compose one group (also denoted as a shift group), the same shift direction is assigned heuristically, in order to cover all kernel dimensions. Moreover, pointwise

convolution is applied before and after the shift-based layer, with the aim of making it invariant to the permutation of input and output channels [67]. This avoids the mapping of each channel to a shift group, which is a very expensive combinatorial problem, in the sense that any arbitrary permutation for the shift kernel can be chosen after setting each shift group (disregarding the channel order) since the two pointwise convolutions allow different permutations of the shift to be equivalent.

With this in mind, a new shift-based neural network for spectral–spatial HSI data classification has been developed by implementing two highly differentiable parts in the proposed network: 1) the feature extractor layers and 2) the classification layers. These layers are explained, in detail, in the following.

1) Shift-Based Feature Extractor for HSI Data Processing: Like any standard CNN for HSI data classification [52], our network has a feature extraction stage that learns different hierarchical and abstract representations obtained from the original input data. The network's input is extracted from a normalized HSI scene by cropping the image into patches of $m \times m$ pixels centered on the target pixel $\mathbf{x}_{m/2+1,m/2+1} \in \mathbb{R}_c^n$, being n_c the number of spectral bands and setting $m = 11$ [41]. Also, with the aim of taking advantage of border pixels, a mechanism for mirroring the HSI scene edges has been implemented [52].

As we can observe in Fig. 3, HSI data patches are sent to the feature extractor. The first group of layers performs a standard $3 \times 3 \times n_c \times 16$ convolution over the input data, followed by batch-normalization [68] and a nonlinear activation function implemented via ReLU [62]. In this sense, the input data is transformed into a more suitable form, being the spatial–spectral data downsampled to $9 \times 9 \times 16$. On the one hand, this reduces the input noise, and on the other hand, we extract compact and abstract features that will become more robust and discriminative as the network is trained, being determinant for the final classification output.

After extracting the first features, three shift blocks are applied. Inspired by residual blocks [58], each shift block follows the pointwise convolution-batch normalization-ReLU-shift-pointwise convolution-batch normalization-ReLU structure, where the input and output volumes are combined through an additive shortcut connection. On this wise, the block's input is first processed by a $1 \times 1 \times n_f^{(l-1)} \times n_f^{(l)}$ convolution to provide invariance to different channel permutations. Then, the shift-based layer reorganizes the spatial information, while the second $1 \times 1 \times n_f^{(l)} \times n_f^{(l)}$ pointwise convolution combines the spectral information across the input channels. Finally, the input and output volumes are combined through the shortcut connection to reuse information, improving the forward step by avoiding data degradation and enhancing the back-propagation by avoiding the vanishing gradient problem. Related to this, it must be noted that the first shift block maintains the same spectral size of the data volumes, i.e., $n_f^{(l-1)} = n_f^{(l)}$, so the shortcut applies an identity function. However, the second and third shift blocks elongate the number of channels (in particular, the first pointwise convolution), i.e., $n_f^{(l-1)} \neq n_f^{(l)}$, so a pointwise convolution layer is included into the shortcut to adapt the number of input channels to the number of output channels.

TABLE I
PROPOSED NETWORK TOPOLOGY

Layer/Block	Parameters	Batch norm	Activation function
1^{st} Conv	$3 \times 3 \times n_c \times 16$	Yes	ReLU
1^{st} Shift block			
Pointwise Conv	$1 \times 1 \times 16 \times 16$	Yes	ReLU
Shift layer	$9 \times 9 \times 16$	-	-
Pointwise Conv	$1 \times 1 \times 16 \times 16$	Yes	ReLU
2^{nd} Shift block			
Pointwise Conv	$1 \times 1 \times 16 \times 32$	Yes	ReLU
Shift layer	$9 \times 9 \times 32$	-	-
Pointwise Conv	$1 \times 1 \times 32 \times 32$	Yes	ReLU
Shortcut Conv	$1 \times 1 \times 16 \times 32$	Yes	-
3^{rd} Shift block			
Pointwise Conv	$1 \times 1 \times 32 \times 64$	Yes	ReLU
Shift layer	$9 \times 9 \times 64$	-	-
Pointwise Conv	$1 \times 1 \times 64 \times 64$	Yes	ReLU
Shortcut Conv	$1 \times 1 \times 32 \times 64$	Yes	-
Average pool	9×9	-	-
2^{st} FC	c	-	Softmax

We can easily modify the behavior of these shift blocks to work as a bottleneck (reducing the number of channels and then increasing them) or an inverted bottleneck (increasing the number of channels and then reducing them) by introducing an expansion rate ε to scale the number of channels in the pointwise convolution layers of every shift block. To achieve this, the first convolution is defined as $1 \times 1 \times n_f^{(l-1)} \times n_f^{(l)} \cdot \varepsilon$, while the second one is in fact defined as $1 \times 1 \times n_f^{(l)} \cdot \varepsilon \times n_f^{(l)}$.

Finally, the resulting output volume is processed and reshaped into a vector form by an average pooling layer [69]. Then, the obtained feature vector is sent to the classifier.

2) Classification Layers for HSI Data Categorization: The adopted classifier is a multilayer perceptron (MLP) composed of one fully connected (FC) layer. The obtained feature volume is reshaped into a vector of 64 elements and sent to the FC layer, which contains c perceptrons, being c the number of classes. The final activation function is the softmax. Table I provides the topology details of the proposed shift-based network for spectral–spatial HSI data classification. Furthermore, the implemented model has been trained by the stochastic gradient descend (SGD) optimizer, employing 200 epochs and a learning rate of 0.01, and with a batch size of 100 samples.

III. EXPERIMENTAL RESULTS

This section demonstrates the benefits of our newly proposed shift-based network for spectral–spatial HSI data classification in terms of both computational performance and accuracy. Focusing on computer and storage requirements, we have measured the the training time required, the number of learning parameters required, and the number of FLOPs consumed by the proposed architecture. In this context, the number of FLOPs is calculated following (1). As the proposed network can be divided into three main parts, i.e., the head (composed of a first convolution layer), the body (composed of three shift-based blocks), and the final classifier (composed of one FC layer), we can obtain the FLOPs of each part as follows. The first convolution unit involves FLOPs_C1, which can be obtained as

$$\text{FLOPs\_C1} = n_c \cdot n_k^{(1)} \cdot n_k^{(1)} \cdot n_f^{(1)} \cdot m^{(1)} \cdot m^{(1)} \tag{6}$$

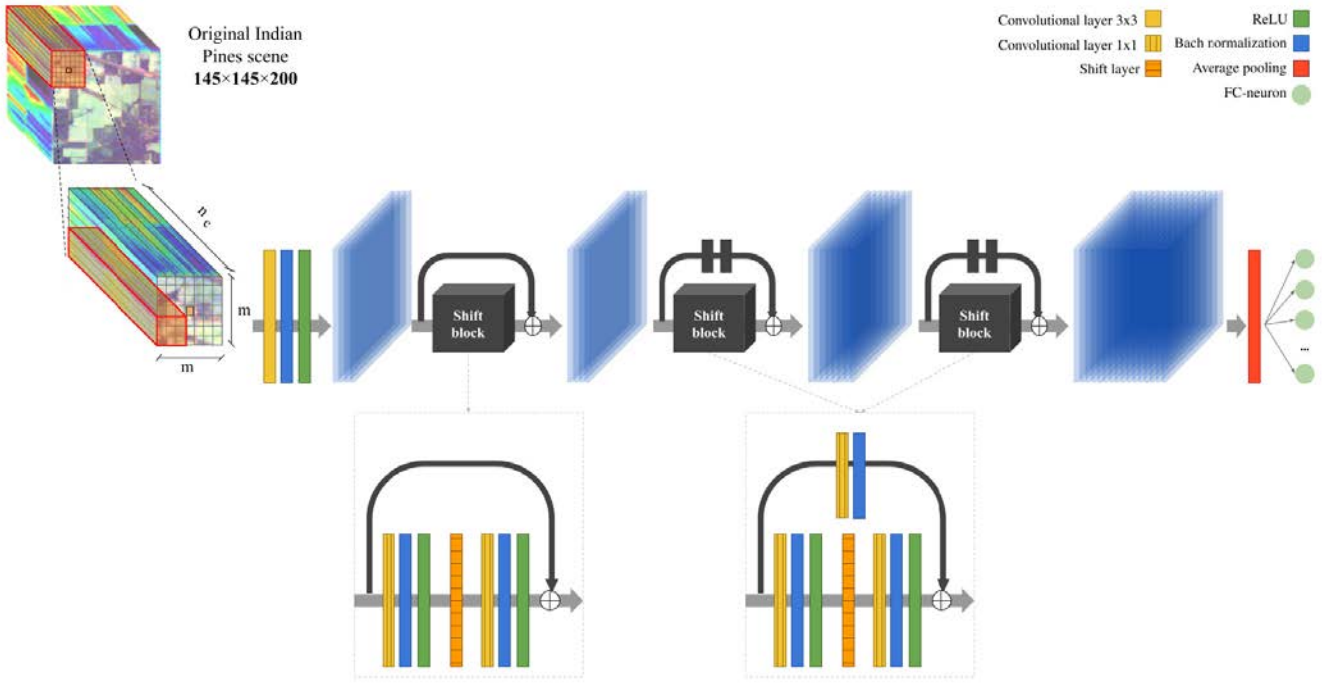


Fig. 3. Graphical overview of the proposed shift-based architecture for HSI data classification.

where n_c is the original number of spectral channels, $n_f^{(1)} = 16$ and $n_k^{(1)} = 3$ define the number of filters and the spatial size of them, respectively, and $m^{(1)} = 9$ denotes the spatial size of the output volume. Then, after the first convolution layer comes the body of the proposed network, which is composed of three shift blocks. Each one comprises two pointwise convolution layers and one shift layer. Moreover, the last two blocks include a shortcut connection, which is also a pointwise convolution. As the shift layer does not introduce FLOPs, only the pointwise convolution layers involve some FLOPs. In addition, these pointwise layers do not affect the spatial size of the feature volumes, so we can simplify $m^{(l)} = 9$ for every convolution layer l in the network. Equation (7) provides the general form to calculate the number of FLOPs of each block

$$\begin{aligned}\text{FLOPs\_B}n \;=\; & n_f^{(l-1)} \cdot 1 \cdot 1 \cdot n_f^{(l)} \cdot 9 \cdot 9 \leftarrow \text{1st Conv.} \\ & + n_f^{(l)} \cdot 1 \cdot 1 \cdot n_f^{(l+1)} \cdot 9 \cdot 9 \leftarrow \text{2nd Conv.} \\ & + \left[n_f^{(l-1)} \cdot 1 \cdot 1 \cdot n_f^{(l+2)} \cdot 9 \cdot 9\right] \\ & \leftarrow \text{Shortcut Conv.} \end{aligned} \tag{7}$$

where FLOPs_Bn identifies the block (where $n = 1, 2, 3$), $n_f^{(l-1)}$ defines the number of input channels, and $1 \cdot 1 \cdot n_f^{(l)}$, $1 \cdot 1 \cdot n_f^{(l+1)}$, and $1 \cdot 1 \cdot n_f^{(l+2)}$ are the kernel sizes of the first, second, and shortcut pointwise convolution (for blocks 2 and 3), respectively. Finally, the number of FLOPs within the FC layer can be obtained as

$$\text{FLOPs\_FC} = \text{dim\_vec} \cdot c \tag{8}$$

where dim_vec $= 64$ is the network output that has been vectorized through the average pooling layer and c is the number of different land cover classes. Once all the FLOPs have been obtained, the number of total FLOPs can be calculated as

$$\begin{aligned}\text{FLOPs} = \; & \text{FLOPs\_C1} + \text{FLOPs\_B1} + \text{FLOPs\_B2} \\ & + \text{FLOPs\_B3} + \text{FLOPs\_FC}. \end{aligned} \tag{9}$$

Also, we can obtain the number of FLOPs of the shift-based blocks (denoted as FLOPsBLOCK) as follows:

$$\text{FLOPsBLOCK} = \text{FLOPs\_B1} + \text{FLOPs\_B2} + \text{FLOPs\_B3}. \tag{10}$$

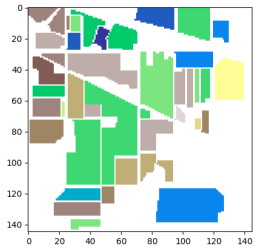
Besides, to evaluate the classification accuracy, the overall (OA), average (AA) accuracies, and kappa coefficient have been considered.

In the following sections, we will explain, in detail, the environment in which the experiments have been conducted. We also describe the HS data sets considered and the obtained results.

A. Experimental Configuration

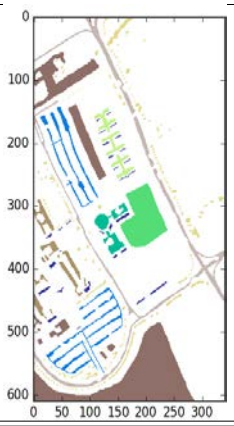
With the aim of testing the performance of the proposed shift-based deep model for spectral–spatial HSI classification, a battery of experiments has been performed on a desktop computer equipped with an X Generation Intel Core i9-9940X processor with 19.25M of Cache and up to 4.40 GHz (14-core/28-way multitask processing), installed over a Gigabyte X299 Aorus, 128 GB of DDR4 RAM. Also, a graphic processing unit (GPU) NVIDIA Titan RTX GPU with 24-GB GDDR6 of video memory and 4608 cores has been employed. The operating system is Ubuntu 18.04.3. In order to efficiently implement the proposed approach, all

INDIAN PINES (IP)



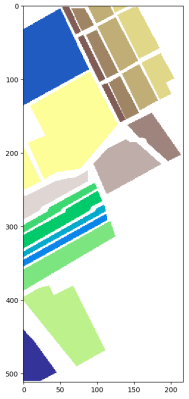
Color	Land-cover type	Samples
	Background	10776
	Alfalfa	46
	Corn-notill	1428
	Corn-min	830
	Corn	237
	Grass/Pasture	483
	Grass/Trees	730
	Grass/pasture-mowed	28
	Hay-windrowed	478
	Oats	20
	Soybeans-notill	972
	Soybeans-min	2455
	Soybean-clean	593
	Wheat	205
	Woods	1265
	Bldg-Grass-Tree-Drives	386
	Stone-steel towers	93
	Total samples	21025

UNIVERSITY OF PAVIA (UP)



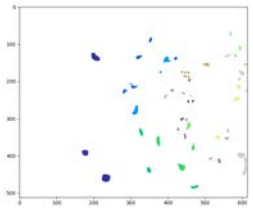
Color	Land-cover type	Samples
	Background	164624
	Asphalt	6631
	Meadows	18649
	Gravel	2099
	Trees	3064
	Painted metal sheets	1345
	Bare Soil	5029
	Bitumen	1330
	Self-Blocking Bricks	3682
	Shadows	947
	Total samples	207400

SALINAS VALLEY (SV)



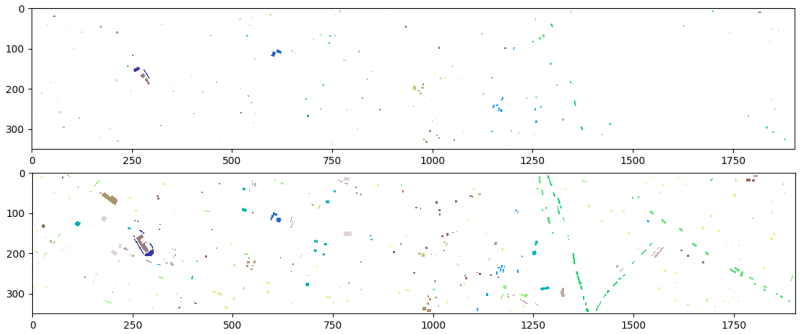
Color	Land-cover type	Samples
	Background	56975
	Brocoli-green-weeds-1	2009
	Brocoli-green-weeds-2	3726
	Fallow	1976
	Fallow-rough-plow	1394
	Fallow-smooth	2678
	Stubble	3959
	Celery	3579
	Grapes-untrained	11271
	Soil-vinyard-develop	6203
	Corn-senesced-green-weeds	3278
	Lettuce-romaine-4wk	1068
	Lettuce-romaine-5wk	1927
	Lettuce-romaine-6wk	916
	Lettuce-romaine-7wk	1070
	Vinyard-untrained	7268
	Vinyard-vertical-trellis	1807
	Total samples	111104

KENNEDY SPACE CENTER (KSC)



Color	Land-cover type	Samples
	Background	309157
	Scrub	761
	Willow-swamp	243
	CP-hammock	256
	Slash-pine	252
	Oak/Broadleaf	161
	Hardwood	229
	Swap	105
	Graminoid-marsh	431
	Spartina-marsh	520
	Cattail-marsh	404
	Salt-marsh	419
	Mud-flats	503
	Water	927
	Total samples	314368

UNIVERSITY OF HOUSTON (UH)



Color	Land cover type	Samples train	Samples test
	Background	649816	
	Grass-healthy	198	1053
	Grass-stressed	190	1064
	Grass-synthetic	192	505
	Tree	188	1056
	Soil	186	1056
	Water	182	143
	Residential	196	1072
	Commercial	191	1053
	Road	193	1059
	Highway	191	1036
	Railway	181	1054
	Parking-lot1	192	1041
	Parking-lot2	184	285
	Tennis-court	181	247
	Running-track	187	473
	Total samples	2832	12197

Fig. 4. Number of available labeled samples in the Indian Pines (IP), University of Pavia (UP), Salinas Valley (SV), and Kennedy Space Center (KSC) HSI data sets.

tested models have been parallelized on the available GPU using Pytorch.

B. HSI Data Sets

In order to test the proposed model on aerial and satellite HSI scenes, five public[4] and widely used HSI data sets have been considered in our experiments: Indian Pines (IP), University of Pavia (UP), Salinas Valley (SV), Kennedy Space Center (KSC), and University of Houston (UH). Fig. 4 shows, for each data set, its corresponding ground-truth information with the number of samples per class. In the following, we summarize the characteristics of each data set.

1) *Indian Pines (IP):* The IP data set was gathered by the airborne visible/infrared imaging spectrometer (AVIRIS) [20] sensor in 1992, and it covers an area comprising different agricultural fields in Northwestern Indiana, USA. This image contains 145 × 145 pixels with a spatial resolution of 20 meters per pixel (mpp) and 224 spectral bands in the wavelength range from

[4]Available online, including the training and test sets, from http://dase.grss-ieee.org

400 to 2500 nm. In our experiments, four null bands and other 20 bands corrupted by the atmospheric water absorption effect have been removed, keeping the remaining 200 bands. The IP ground truth contains a total of 16 mutually exclusive land cover classes.

2) *Salinas Valley (SV):* The SV image was also captured in 1998 by the AVIRIS instrument over the agricultural land of Salinas Valley in California, USA. The data comprise of 512 × 217 pixels with a spatial resolution of 3.7 mpp. As for the IP data set, the water absorption bands, i.e., channels from 108th to 112th and from 154th to 167th, together with the 224th band, have been discarded. A total of 16 different land cover classes are included in the SV ground-truth data.
3) *Kennedy Space Center (KSC):* As IP and SV scenes, the KSC image was 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 data set was gathered by the Reflective Optics System Imaging Spectrometer (ROSIS) sensor [70] in 2001 over the University of Pavia, Northern Italy. This image contains 103 spectral bands ranging from 430 to 860 nm 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 nine different class labels.
5) *University of Houston (UH):* The UH scene [71] was acquired by the Compact Airborne Spectrographic Imager (CASI) sensor [72] over the Houston University campus in June 2012, collecting spectral–spatial information from an urban area. This scene comprises 114 bands and 349 × 1905 pixels with wavelengths ranging from 380 to 1050 nm. Twenty-one principal components have been considered during classification tasks. Its ground-truth information comprises 15 different land cover classes, providing two spatial-disjoint subsets of training and testing samples.

C. Performance Evaluation

With the aim of evaluating the computational performance obtained by the proposed shift-based network for spectral–spatial HSI data classification in terms of training times, number of learnable parameters and FLOPs, and to make a thorough analysis of the implemented deep learning architecture in terms of accuracy, several experiments have been conducted, considering for each one of the five Monte Carlo runs.

1) The first experiment focuses on the classification accuracy obtained by the proposed shift-based network compared with a standard ResNet and a reduced-parameter ResNet (ResNetR3) when several training parameters are considered; in particular, the OA evolution has been measured by considering 1%, 3%, 5%, 10%, and 15% of the available labeled samples per class for the IP, SV, KSC, and UP scenes. In this context, ResNet and ResNetR3 models have been implemented with the same architecture as the shift-based network by replacing the shift blocks with their residual counterparts, which comprises of two 3 × 3 convolution layers with zero-padding to maintain the same spatial dimensions along with the convolution layers. As the standard ResNet contains a large number of parameters in comparison with the shift-based network, we reduce the ResNetR3 parameters in every convolution layer by a factor $R = 3$ in order to approximately match the number of parameters in the proposal. In particular, the number of filters of each layer has been reduced by $n_f^{(l)}/R$. It must be noted that $\varepsilon = 1$ is considered for the shift-based network.
2) The second experiment extracts more information about the classification performance obtained by the proposed network, the standard ResNet, and the reduced ResNetR3 when processing HSI data. In this sense, the classification per class, OA, AA, and kappa coefficient have been measured for the IP (with 5% of training data), KSC (5%), SV (1%), UP (1%), and UH scenes. Moreover, to measure the computational and memory consumption of the considered models, the training time, the number of FLOPs, and the number of learnable parameters have been obtained for both the entire architecture and the body, composed only by the three shift and residual blocks, with the aim of specifically examining the improvement of the proposal.
3) The third experiment conducts a more focused study on the expansion ε and reduction R parameters of shift-net and ResNetRx (being x the reduction value), respectively, in terms of the number of parameters and FLOPs, together with their impact on the model's OA. In this regard, the IP (with 5% of training data), KSC (5%), SV (1%), UP (1%), and UH scenes have been considered.
4) The fourth experiment makes a comparison of the proposed method with some current state-of-the-art techniques by considering multiple spatial sizes for the input patches, i.e., 5 × 5, 7 × 7, 9 × 9, and 11 × 11. In particular, the proposed shift-based network has been compared with five spectral–spatial networks for HSI data classification: the spectral–spatial ResNet (SSRN) [57], the pyramidal ResNet (P-RN) [58], the densely connected ResNet (DenseNet) [73], the dual-path network (DPN) [60], and the capsule network (CapsNet) [53]. In this experiment, the IP (with 20% of training data), KSC (20%), and UP (10%) scenes have been considered.

1) Experiment 1: OA Evolution With Different Training Percentages: In order to explore the performance of the proposed model when different amounts of training samples are available, this experiment analyzes the impact over the resulting OA when different percentages of the available labeled data are employed to train the model. Related to this, four HSI scenes have been considered for this experiment in order to ensure that our model can work with different types of images and spectral–spatial resolutions. In particular,

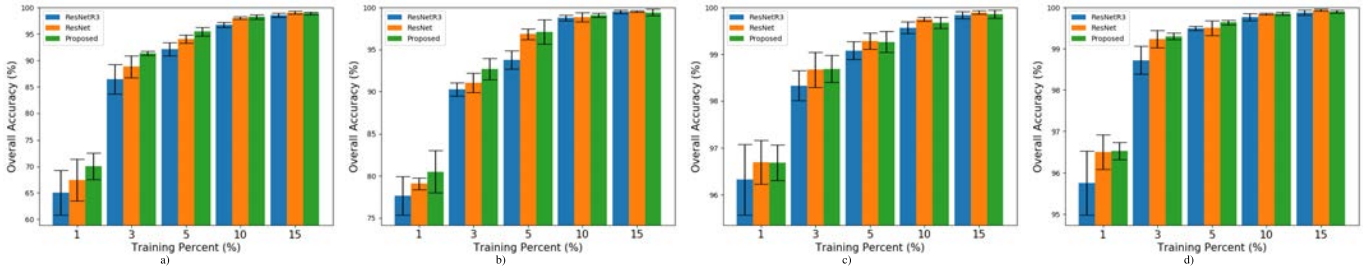


Fig. 5. Classification results obtained by the proposed shift-based network, the ResNet, and the ResNetR3 considering four real HSI scenes with different amounts of training samples: (a) IP, (b) KSC, (c) SV, and (d) UP.

TABLE II

CLASSIFICATION RESULTS OBTAINED BY SHIFT-BASED NETWORK, STANDARD RESNET, AND REDUCED-PARAMETER RESNETR3 USING 5% OF THE AVAILABLE LABELED DATA FOR TRAINING WITH THE IP AND KSC SCENES AND 1% FOR TRAINING WITH THE SV AND UP SCENES. ALSO, THE FIXED TRAINING SAMPLES PROVIDED FOR THE UH SCENE ARE USED. THE INPUT SPATIAL PATCH SIZE HAS BEEN SET TO 11×11 IN ALL CASES (PARAMETERSBLOCKS AND PARAMETERS DEFINE THE NUMBER OF PARAMETERS CONSIDERING ONLY THE THREE SHIFT/RESIDUAL-BASED BLOCKS AND THE ENTIRE MODEL, RESPECTIVELY. SIMILARLY, FLOPSBLOCKS AND FLOPS PROVIDE THE NUMBER OF FLOPS CONSIDERING ONLY THE THREE SHIFT/RESIDUAL-BASED BLOCKS AND THE ENTIRE MODEL, RESPECTIVELY)

Class	DATASETS														
	Indian Pines			**Kennedy Space Center**			**Salinas Valley**			**University of Pavia**			**University of Houston**		
	ResNetR3	ResNet	Proposed	ResNetR3	ResNet	Proposed	ResNetR3	ResNet	Proposed	ResNetR3	ResNet	Proposed	ResNetR3	ResNet	Proposed
1	47.73	45.45	**51.82**	**99.86**	99.59	99.81	99.86	**99.90**	99.62	95.94	95.86	**97.17**	81.52	81.69	**82.66**
2	91.07	92.90	**93.84**	75.67	87.27	**92.12**	99.17	**99.80**	99.71	99.20	**99.20**	99.12	83.85	84.32	**84.40**
3	90.09	95.56	**96.09**	93.83	95.47	**96.87**	**99.25**	98.26	98.93	79.78	**87.86**	85.09	97.19	**99.13**	98.93
4	84.71	91.38	**92.62**	81.84	**87.03**	83.85	96.96	97.43	**98.91**	96.00	**96.13**	95.19	84.64	87.18	**88.28**
5	89.98	90.63	**94.03**	65.75	75.03	**76.08**	98.50	**98.80**	96.12	99.07	98.35	**99.19**	99.76	99.36	**99.77**
6	96.30	97.26	**97.89**	**95.04**	93.03	91.38	99.69	99.63	**99.79**	94.07	**95.67**	94.05	88.81	91.47	**92.17**
7	56.30	65.19	**73.33**	94.00	97.80	**98.80**	99.06	99.57	**99.60**	83.07	**89.86**	87.36	**82.07**	81.23	79.16
8	97.49	99.43	**99.78**	97.41	97.51	**99.76**	92.03	92.26	**93.09**	93.53	92.46	**95.62**	72.31	76.73	**79.77**
9	71.58	**73.68**	67.37	95.63	98.74	**99.39**	99.75	**99.86**	99.43	92.06	95.37	**96.50**	77.58	80.62	**81.38**
10	91.33	90.56	**93.28**	97.14	98.44	**98.54**	96.73	96.94	**97.68**	-	-	-	61.31	62.98	**76.52**
11	95.01	96.77	**97.59**	99.35	**99.35**	99.30	**96.61**	91.07	96.24	-	-	-	91.80	96.03	**96.79**
12	82.10	86.68	**89.77**	96.15	**99.87**	98.20	**99.86**	98.95	99.75	-	-	-	93.50	95.89	**97.08**
13	89.85	96.62	**97.23**	100.0	100.0	100.0	98.52	99.12	**99.87**	-	-	-	**83.30**	78.11	75.02
14	95.46	**97.55**	97.55	-	-	-	97.68	98.15	**98.26**	-	-	-	99.92	100.0	100.0
15	91.77	88.23	**94.12**	-	-	-	90.86	**93.58**	91.96	-	-	-	**97.42**	92.90	93.07
16	89.09	91.82	**99.32**	-	-	-	**97.24**	96.89	96.10	-	-	-	-	-	-
OA	92.11	94.07	**95.47**	94.97	96.86	**97.10**	96.32	**96.70**	96.69	95.76	96.51	**96.53**	84.44	85.78	**87.40**
AA	84.99	87.48	**89.73**	91.67	94.55	**94.93**	97.61	97.51	**97.82**	92.52	**94.53**	94.37	86.33	87.17	**88.33**
K(x100)	91.00	93.24	**94.83**	94.40	96.50	**96.77**	95.91	**96.32**	96.31	94.36	95.37	**95.39**	83.14	84.57	**86.32**
Parameters	41308	106800	**41264**	39253	103149	**37613**	**43252**	110256	44720	33192	92377	**26841**	26772	80959	**15423**
ParametersBLOCKs	24498	76928	**11392**	24498	76928	**11392**	24498	76928	**11392**	24498	76928	**11392**	24498	76928	**11392**
FLOPs	3267954	8513152	**3204736**	3110382	8233024	**2924608**	**3425418**	8793088	3484672	2631285	7381296	**2072880**	2093499	6425232	**1116816**
FLOPsBLOCKs	1955178	6179328	**870912**	1955178	6179328	**870912**	1955178	6179328	**870912**	1955178	6179328	**870912**	1955178	6179328	**870912**
Tr. time (s)	7.36	**7.20**	8.19	**2.71**	2.77	2.90	7.31	**7.20**	7.61	**7.30**	7.41	7.88	**43.08**	44.80	52.94

the IP, UP, KSC, and SV scenes have been used with 1%, 3%, 5%, 10%, and 15% of labeled samples per class for training (and the remaining samples used for testing). In addition, three networks with similar architectures have been tested: the proposed shift-based network with $\varepsilon = 1$, a standard ResNet, and a reduced-parameter ResNetR3 (with reduction factor $R = 3$).

Fig. 5 shows the obtained results after five Monte Carlo iterations. We can observe a quite similar behavior between the different data sets. When we consider a training percentage of 1%, the reduced-parameter ResNetR3 achieves the worst OA result, while the proposed shift-based network and the ResNet are able to reach the best results. We can particularly highlight the result obtained by our shift-based network with the IP and KSC scenes [see Fig. 5(a) and (b)]. This is quite interesting since these two HSI scenes are particularly complex to classify due to their lower spatial resolution, which leads to more highly mixed spectral signatures than, for example, the UP image [see Fig. 5(d)]. It should be noted that our proposal implements a model whose number of parameters is significantly lower than the standard ResNet; however, it is able to achieve similar or even better classification results, being much more robust than the standard ResNet when few training samples are employed.

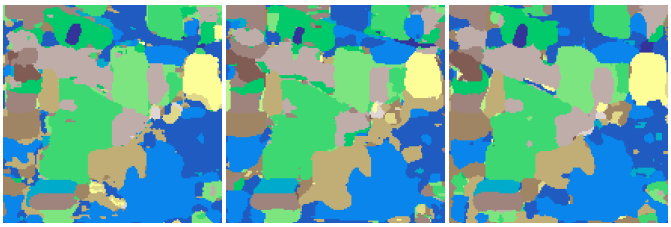
Furthermore, the OA of the three networks improves by increasing the number of training samples, being the shift-based network the model that reaches the best OA results in almost all cases, achieving very close results to those obtained by the ResNet in those cases in which the proposal is not the best. As expected, the three models achieve similar results with the highest training percentages.

2) Experiment 2: Classification Performance Versus Computing Requirements: The second experiment follows the previous experiment, considering also five Monte Carlo iterations and exploring the classification accuracy reached by the considered models in each land-cover class, in addition to the OA, AA, and kappa coefficient values. Moreover, this experiment shows the training times of each deep network, the number of parameters required by the models, and the number of FLOPs executed. In this context, Table II provides both the

TABLE III

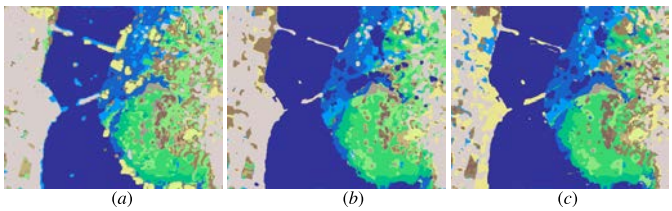
ACCURACY PERFORMANCE IN TERMS OF OA, AA, AND KAPPA VALUES WHEN CONSIDERING SPATIAL DISJOINT DATA SETS FOR THE PROPOSED NETWORK, STANDARD RESNET, AND RESNETR3

	Indian Pines				University of Pavia				University of Houston			
Method	OA	AA	K(x100)	Tr. time (s)	OA	AA	K(x100)	Tr. time (s)	OA	AA	K(x100)	Tr. time (s)
ResNet	80.10	68.51	77.33	85.01	91.71	89.38	88.69	71.02	84.44	86.33	83.14	44.80
ResNetR3	86.93	74.45	85.11	83.08	91.98	90.68	89.10	70.39	85.78	87.17	84.57	43.08
Proposed	**89.80**	**78.96**	**88.39**	88.01	**93.59**	**92.60**	**91.30**	76.25	**87.40**	**88.33**	**86.32**	52.94



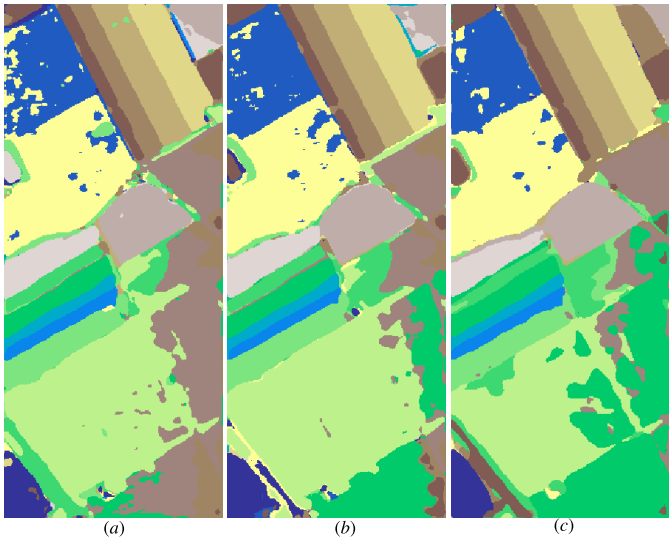
(*a*) (*b*) (*c*)

Fig. 6. Classification maps obtained for the IP scene (using 5% of the available labeled samples). The obtained OAs are shown in brackets. (a) ResNetR3 (92.11%). (b) ResNet (94.07%). (c) Proposed (95.47%).



(*a*) (*b*) (*c*)

Fig. 7. Classification maps obtained for the KSC scene (using 5% of the available labeled samples). The obtained OAs are shown in brackets. (a) ResNetR3 (94.97%). (b) ResNet (96.86%). (c) Proposed (97.10%).



(*a*) (*b*) (*c*)

Fig. 8. Classification maps obtained for the SV scene (using 1% of the available labeled samples). The obtained OAs are shown in brackets. (a) ResNetR3 (96.32%). (b) ResNet (96.70%). (c) Proposed (96.69%).

parameters and the FLOPs of the entire set of models (which have been denoted as Parameters and FLOPs, respectively), as well as those that correspond with the shift-based and residual blocks (designated as ParametersBLOCKs and FLOPsBlocks, respectively), i.e., without the first convolution layer and the FC layer that composes the classifier.

Focusing on the IP scene, we can observe the obtained classification results per land-cover class in Table II, being the ones obtained by the proposed model higher than those achieved when using the standard ResNet and the reduced-parameter ResNetR3 models. Also, the OA, AA, and kappa values achieved by the shift-based network are the best: the OA is 1.4% and 3.36% points higher than that of ResNet and ResNetR3, respectively; the AA is 2.25% and 4.74% points higher, respectively, and the kappa is also 1.59 and 3.83 points better. This behavior is repeated in other challenging HSI scenes, such as the KSC and UH. In particular, the OA obtained by the proposed model for the KSC is 0.24 points better than that of ResNet and 2.13 points higher than that of ResNetR3, while, for the UH scene, it is 1.62 and 2.96 points better, respectively. In this sense, we can conclude that both the ResNet and the ResNetR3 models are significantly affected by the complexity of these three scenes.

On the contrary, for less challenging HSI scenes, such as UP and SV (with spectral information that is less spectrally mixed and with higher spatial resolution, which can improve decisively the classification task [41]), the ResNet and ResNetR3 models improve their results. Focusing on the SV scene, the ResNet is able to reach the best OA (96.7) and kappa (96.32) values; however, the shift-based network (which shows the best AA value—97.82) is able to reach very similar values, being its OA and kappa only 0.01 points lower than ResNet. With the UP scene, the ResNet achieves the best AA (94.53), being only 0.16 points better than the proposed network, which still exhibits the best OA (96.53) and kappa (95.39) values. These results are depicted graphically in Figs. 6–10, where the classification maps provided by each model are reported. The three networks provide typical spatial–spectral model maps, with the borders between classes generally well defined and without salt and pepper noise. It is noteworthy that the proposed network and ResNet achieve similar results in some images (for instance, the UH), while, in other images, the proposed network achieves a much more accurate classification in certain complicated regions of the scenes (particularly, for the IP and KSC scenes).

In addition to these classification results, which have been obtained by a random selection of training samples, Table III provides the accuracy performance of these three networks considering the IP, UP, and UH scenes without spatial overlapping between the training and test sets (the so-called spatial disjoint images). A reduction in accuracy is observed for all methods on IP and UP images, either because the spatial information from the test is no longer available in training samples or because these samples are more complex

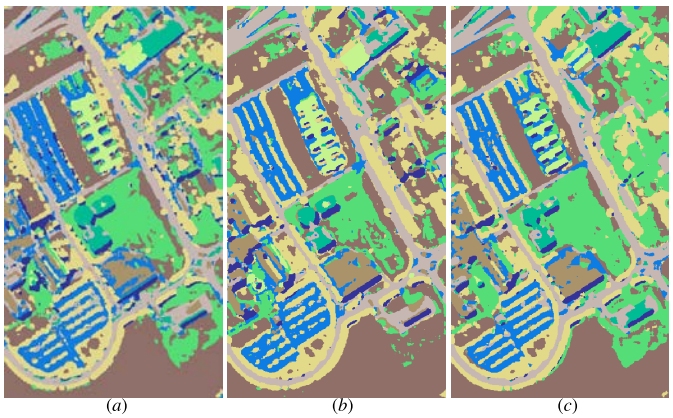


Fig. 9. Classification maps obtained for the UP scene (using 1% of the available labeled samples). The obtained OAs are shown in brackets. (a) ResNetR3 (95.76%). (b) ResNet (96.51%). (c) Proposed (96.53%).

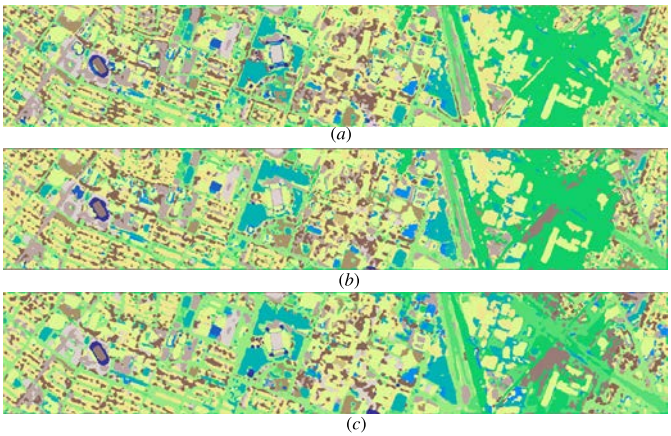


Fig. 10. Classification maps obtained for the UH scene (using the available fixed training set). The obtained OAs are shown in brackets. (a) ResNetR3 (84.44%). (b) ResNet (85.78%). (c) Proposed (87.40%).

to classify. However, this degradation is less pronounced in the proposed network compared with standard ResNet: in particular, ResNet's OA drops 13.97 points in IP and 4.8 in UP, while ResNetR3's OA falls 5.18 points in IP and 3.78 in UP, and the proposed decreases its OA 5.67 points in IP and 2.94 in UP. Furthermore, the proposed network achieves the best classification results in terms of OA, AA, and kappa measurements.

At this point, it is important to emphasize that, considering Table II, both ResNet and our shift-based model achieve very similar results in terms of accuracy; however, the proposed network requires significantly fewer parameters than the standard ResNet. In particular, considering the whole set of models, the proposed shift-based network needs 65 536 fewer parameters than the ResNet in each scene. Focusing on the shift and residual blocks, the proposed model has to learn 11 392 parameters to perform correctly, while the ResNet needs 76 928 parameters to reach its classification performance, that is, the shift blocks are able to reduce 65 536 parameters in comparison with the residual blocks. This means that the shift-based network consumes significantly less memory than the standard ResNet to store the parameters while reducing the number of FLOPs when applying the convolutional kernels. In particular, the proposed network consumes 5 308 416 FLOPs less than the standard ResNet in every scene. Focusing on the IP scene, if we compare the FLOPs consumed by the entire proposed network (3 204 736) with those consumed by the shift blocks (870 912), we can determine that the first convolution layer and the two FC layers consume 2 333 824 FLOPs, i.e., they consume 2.68× more than the three shift blocks. On the contrary, in the ResNet model, the three residual blocks (6 179 328 FLOPs) consume 2.65× more FLOPs than the first convolution layer and the FC layers together (2 333 824 FLOPs), so the shift block consumes significantly less than a residual block (7.10× less).

In order to make a more fair comparison between neural models with the same number of parameters and FLOPs, we have implemented the ResNetR3 with a parameter reduction of $R = 3$, where the number of filters of each layer has been reduced by $n_f^{(l)}/R$. In this sense, the entire ResNetR3 requires, on average, 61 952.8 parameters less than the standard ResNet. With this significant reduction, we can observe that the number of parameters of the entire ResNetR3 model is closer to the shift-based network than to the standard ResNet, being the number of parameters even smaller than that of the shift-based network with the SV scene. However, despite the large reduction in parameters applied in all the convolution layers, if we focus only on the residual blocks, we can clearly see that they need to adjust 13 106 more parameters than the proposed block, i.e., 2.15× more parameters. This will affect the number of FLOPs executed by the residual blocks of the ResNetR3 model. In particular, comparing the FLOPs of the whole set of considered models, ResNetR3 seems to run fewer FLOPs than the shift-based network (59 254 FLOPs less). This is because the reduction factor is also applied to the first convolution layer, considerably reducing the number of parameters and operations to be performed on the data. However, if we focus on the residual blocks, the ResNetR3 consumes 1 084 266 more FLOPs than the proposed model, i.e., 2.25× more FLOPs. In addition, despite having a similar architecture with a comparable number of parameters and FLOPs, the classification results of ResNetR3 are significantly worse than those achieved by the proposed network.

Finally, we show the training times of each model for each HSI data set. As we can observe, our model is slightly slower than the ResNet and ResNetR3 models. This may be because, all three models have been optimized on GPUs, so the arithmetic operations (the matrix multiplication between weight and data) have been properly parallelized on the device, while the shift operation has to fix the number of channels for each shift direction (in order to create the shift groups) and then move the data of each channel across the selected spatial direction. These results suggest that these memory operations should be optimized to reduce computational times.

These results strongly support the proposed model since it is not only able to reduce the number of required parameters in comparison to traditional (spatial-based) convolutional layers—which significantly reduces both memory consumption and execution time by reducing also the number of operations to be executed on the HSI data—but is also able

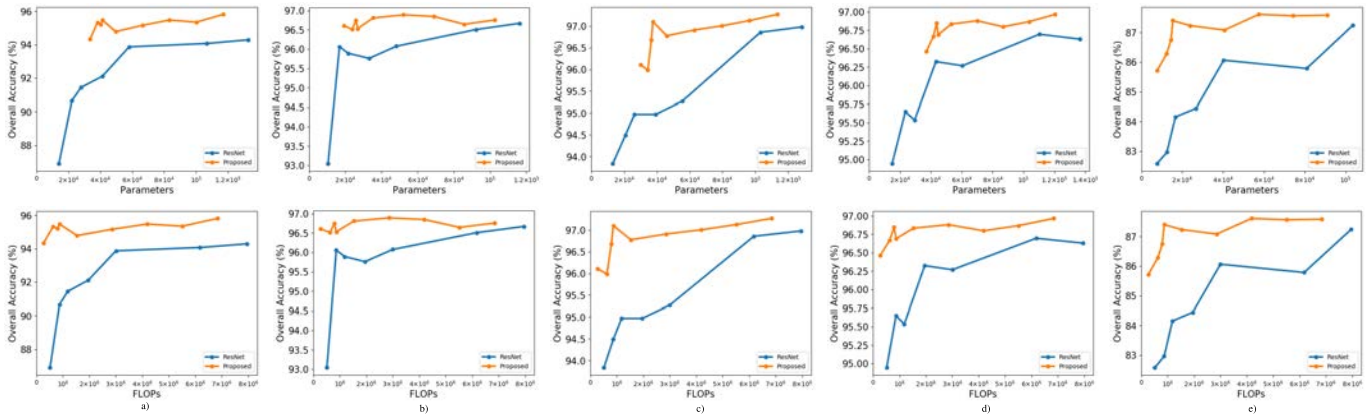


Fig. 11. Overall accuracies (OAs)—obtained by the proposed shift-based network and the equivalent reduced-parameter ResNet—versus the number of required (top) parameters and (bottom) FLOPS for the considered scenes: (a) IP, (b) UP, (c) KSC, (d) SV, and (e) UH.

to match or even outperform the accuracy results of complex models, such as the ResNet and the reduced-parameter ResNetR3.

3) Experiment 3: Comparison Between the Expansion Rate ε and the Reduction Factor R: Related to the last part of the previous experiment, we now delve into the relationship between the shift-based network (with a given expansion rate ε) and its—similar—residual model (with the corresponding reduction factor R) by carefully monitoring the effect of the number of parameters and FLOPs of the blocks on the achieved OA. The models have been trained by considering 3% of the available labeled samples from the UP and SV scenes and 5% of the available labeled samples from the IP and KSC images. For the UH scene, we use the labeled samples from the available (spatially disjoint) training–test set to adjust the model parameters.

To obtain the shift-based models with different numbers of parameters (and therefore of FLOPs), a search of ε in the range $[0.1, 10]$ has been conducted, implementing both a typical bottleneck block (that first strangles the spectrum and then expands it) and an inverted bottleneck block (that first expands the spectrum and then reduces it). Also, to implement the equivalent reduced-parameter version of the ResNet, a search of R in the range $[1, 11]$ has been carried out. The obtained results are reported in Fig. 11. In general, for both models, the higher the number of parameters, the better the classification result. However, we must note that ResNet suffers from a greater impact than the shift-based network when the number of parameters is lower. On the contrary, the proposed network is able to maintain a relatively good OA in almost every scene, dropping just over one percentage point behind the model with the highest number of parameters and the model with the lowest one. Moreover, it is very interesting to emphasize that the proposed model is able to achieve very good OA values with approximately $3\times$ fewer parameters than the equivalent ResNet, being also $6\times$ faster. Also, focusing on the number of FLOPs, our shift-based network obtains an OA that is approximately $3\times$-$4\times$ higher than the one achieved by ResNet using the same number of FLOPs (the improvement is up to $7\times$ in the IP scene with the lowest number of FLOPs). These results confirm our introspection that, for the same number of parameters/FLOPs, our network is able to learn better and provide higher accuracy scores than conventional ResNets.

TABLE IV

OVERALL ACCURACY (%) ACHIEVED BY DIFFERENT APPROACHES WHEN CONSIDERING DIFFERENT SIZES OF THE INPUT SPATIAL PATCHES. FOR EACH MODEL, A PARAMETER ESTIMATION HAS BEEN CONDUCTED IN ORDER TO PROVIDE AN OVERVIEW OF THE DIFFERENT ARCHITECTURES

Indian Pines						
Spatial Size	**SSRN** [57]	**P-RN** [58]	**DenseNet** [73]	**DPN** [60]	**CapsNet** [53]	**Proposed**
5×5	92.83	98.80	97.85	97.53	97.79	96.83
7×7	97.81	99.26	99.24	99.29	99.30	98.81
9×9	98.68	99.64	99.58	99.64	99.67	99.36
11×11	98.70	99.82	99.74	99.67	99.74	99.50
University of Pavia						
Spatial Size	**SSRN** [57]	**P-RN** [58]	**DenseNet** [73]	**DPN** [60]	**CapsNet** [53]	**Proposed**
5×5	98.72	99.52	99.13	99.21	99.13	99.02
7×7	99.54	99.81	99.71	99.70	99.75	99.56
9×9	99.57	99.79	99.73	99.88	99.73	99.78
11×11	99.79	99.92	99.93	99.94	99.93	99.83
Kennedy Space Center						
Spatial Size	**SSRN** [57]	**P-RN** [58]	**DenseNet** [73]	**DPN** [60]	**CapsNet** [53]	**Proposed**
5×5	98.72	99.52	99.13	99.21	99.13	97.40
7×7	99.54	99.81	99.71	99.70	99.75	99.07
9×9	99.57	99.79	99.73	99.88	99.73	99.56
11×11	99.79	99.92	99.93	99.94	99.93	99.83
Parameters	360K.	2.4M.	1.7M.	370K.	9.0M.	**30K.**

4) Experiment 4: Comparison With Other State-of-the-Art Models Considering Different Spatial Sizes for the Input Patch: Our last experiment evaluates the performance of the proposed network when different spatial sizes are considered for the input patch. In this sense, input patches with sizes $m = \{5, 7, 9, 11\}$ have been considered, employing 20% of the available labeled samples per class for the IP and KSC scenes and 10% of the available labeled samples per class for the of UP image. Furthermore, the obtained results (in terms of OA and number of parameters) have been compared with some widely used spectral–spatial deep models that, in fact, constitute the current state of the art in the field: SSRN [57], P-RN [58], DenseNet [73], DPN [60], and CapsNet [53].

Table IV reports the obtained results. As we can observe, the proposed model improves its classification results when

larger spatial patches are considered, as the other models do. In addition, its accuracy results are very similar to those of the current state-of-the-art methods, reaching even higher accuracies than the SSRN when classifying the IP and UP scenes. However, as we can observe in the last row of every scene, the number of parameters needed by our model (30K) is considerably smaller than that required by the other tested models. In particular, CapsNet (9.0M) and DenseNet (1.7M) are the "heaviest" models, requiring millions of parameters. They are followed by the DPN (370K) and SSRN (360K) models. This implies that our model requires $12\times$ fewer parameters than the thinnest model in the current state of the art, i.e., the SSRN. These results confirm the fact that the proposed method can achieve accuracy scores that are comparable to those achieved by the most widely used models for spatial–spectral classification of HSI images, but requiring a much lower amount of parameters to be adjusted (and of operations to be executed), which leads to significant savings in terms of storage and computational resources.

IV. Conclusion

This work proposes a new deep neural network architecture for spectral–spatial classification of HSIs based on a more efficient building block in terms of the number of parameters and FLOPs. Such building block comprises a shift operation interleaved with pointwise convolutions, where the shift operation moves each channel of its input feature volume in a different spatial direction, while the pointwise convolutions provide invariance to channel permutations. The idea behind this innovative design is to avoid the use of the traditional $n_k \times n_k$ spatial convolutions to process the spatial information contained in the remotely sensed HSI scene because these operations are very expensive in computational terms, requiring the adjustment and learning of a large number of parameters, which must be applied to the input data as a windowing algorithm. As a consequence, deep networks exhibit a great computational load, in addition to rapidly tending to overadjustment due to the large number of parameters to be trained and the scarcity of tagged hyperspectral data. In this context, our shift-based network replaces such costly spatial convolution operations by a two-step process: first, the shift operation moves the input channels following different spatial directions in order to mix and combine the spatial information of each channel, and then, the pointwise convolution mixes and combines the spectral information along the feature volume channels. Opposite to traditional spatial convolution, the shift operation does not involve parameters to be learned or FLOPs to be executed, being limited to a simple adjustment of data in memory, while the cost of the 1×1 pointwise convolution is noticeably less than that of a standard $n_k \times n_k$ convolution. This provides a new way to, on the one hand, reduce the computational burden of deep models when facing HSI data classification tasks and, on the other hand, to deal with the overfitting problem by working with fewer parameters.

Our experiments, conducted over five widely used HSI scenes, demonstrate that the proposed method is able to match or even improve the classification results of the equivalent residual equivalent model but employing significantly fewer parameters and executing much fewer operations per block. Moreover, the obtained results demonstrate that the proposed method is able to achieve classification accuracies that are very similar to those obtained by state-of-the-art spatial–spectral classifiers for HSI images but reducing the computation cost significantly compared with the five considered (widely used) spectral–spatial networks, requiring a significantly lower number of parameters to be adjusted. This allows our proposal to consume less storage and computational resources.

As future works, we would like to introduce the shift mechanism in other different state-of-the-art networks in computer vision and remote sensing image processing and also improve our network efficiency by reducing the cost of moving data into memory.

References

[1] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.

[2] Y. Liu, N. V. Chawla, M. P. Harper, E. Shriberg, and A. Stolcke, "A study in machine learning from imbalanced data for sentence boundary detection in speech," *Comput. Speech Lang.*, vol. 20, no. 4, pp. 468–494, Oct. 2006.

[3] S. Casale, A. Russo, G. Scebba, and S. Serrano, "Speech emotion classification using machine learning algorithms," in *Proc. IEEE Int. Conf. Semantic Comput.*, Aug. 2008, pp. 158–165.

[4] L. Deng and X. Li, "Machine learning paradigms for speech recognition: An overview," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 21, no. 5, pp. 1060–1089, May 2013.

[5] F. Sebastiani, "Machine learning in automated text categorization," *ACM Comput. Surv.*, vol. 34, no. 1, pp. 1–47, Mar. 2002.

[6] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 649–657.

[7] R. S. Michalski *et al.*, *Machine Learning and Data Mining: Methods and Applications*, vol. 388. New York, NY, USA: Wiley, 1998.

[8] T. M. Mitchell, "Machine learning and data mining," *Commun. ACM*, vol. 42, no. 11, pp. 30–36, 1999.

[9] A. E. Hassanien and D. A. Oliva, *Advances in Soft Computing and Machine Learning in Image Processing*, vol. 730. New York, NY, USA: Springer, 2017.

[10] A. Asokan and J. Anitha, "Machine learning based image processing techniques for satellite image analysis—A survey," in *Proc. Int. Conf. Mach. Learn., Big Data, Cloud Parallel Comput. (COMITCon)*, 2019, pp. 119–124.

[11] N. Sebe, I. Cohen, A. Garg, and T. S. Huang, *Machine Learning in Computer Vision*, vol. 29. Dordrecht, The Netherlands: Springer, 2005.

[12] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Comput. Intell. Neurosci.*, vol. 2018, pp. 1–13, Feb. 2018.

[13] D. J. Lary, A. H. Alavi, A. H. Gandomi, and A. L. Walker, "Machine learning in geosciences and remote sensing," *Geosci. Frontiers*, vol. 7, no. 1, pp. 3–10, 2016.

[14] G. Camps-Valls, J. Bioucas-Dias, and M. Crawford, "A special issue on advances in machine learning for remote sensing and geosciences [from the guest editors]," *IEEE Geosci. Remote Sens. Mag.*, vol. 4, no. 2, pp. 5–7, Jun. 2016.

[15] A. E. Maxwell, T. A. Warner, and F. Fang, "Implementation of machine-learning classification in remote sensing: An applied review," *Int. J. Remote Sens.*, vol. 39, no. 9, pp. 2784–2817, May 2018.

[16] G. Cheng *et al.*, "Object detection in remote sensing imagery using a discriminatively trained mixture model," *ISPRS J. Photogramm. Remote Sens.*, vol. 85, pp. 32–43, Nov. 2013.

[17] J. Han, D. Zhang, G. Cheng, L. Guo, and J. Ren, "Object detection in optical remote sensing images based on weakly supervised learning and high-level feature learning," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 6, pp. 3325–3337, Jun. 2015.

[18] R. DeFries, "Multiple criteria for evaluating machine learning algorithms for land cover classification from satellite data," *Remote Sens. Environ.*, vol. 74, no. 3, pp. 503–515, Dec. 2000.

[19] A. F. Goetz, G. Vane, J. E. Solomon, and B. N. Rock, "Imaging spectrometry for Earth remote sensing," *Sci.*, vol. 228, no. 4704, pp. 1147–1153, 1985.

[20] G. Vane, R. O. Green, T. G. Chrien, H. T. Enmark, E. G. Hansen, and W. M. Porter, "The airborne visible/infrared imaging spectrometer (AVIRIS)," *Remote Sens. Environ.*, vol. 44, nos. 2–3, pp. 127–143, 1993.

[21] C. H. Chen and P.-G. P. Ho, "Statistical pattern recognition in remote sensing," *Pattern Recognit.*, vol. 41, no. 9, pp. 2731–2741, Sep. 2008.

[22] 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 Geosci. Remote Sens. Mag.*, vol. 1, no. 2, pp. 6–36, Jun. 2013.

[23] F. Melgani and L. Bruzzone, "Classification of hyperspectral remote sensing images with support vector machines," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 8, pp. 1778–1790, Aug. 2004.

[24] J. Ham, Y. Chen, M. M. Crawford, and J. Ghosh, "Investigation of the random forest framework for classification of hyperspectral data," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 3, pp. 492–501, Mar. 2005.

[25] E. Merényi, J. V. Taranik, T. Minor, W. H. Farrand, and R. Green, "Quantitative comparison of neural network and conventional classifiers for hyperspectral imagery," in *Proc. Summaries 6th Annu. JPL Airborne Earth Sci. Workshop*, vol. 1, Pasadena, CA, USA, 1996, pp. 171–174.

[26] M. Pal, "Multinomial logistic regression-based feature selection for hyperspectral data," *Int. J. Appl. Earth Observ. Geoinformation*, vol. 14, no. 1, pp. 214–220, Feb. 2012.

[27] A. Plaza, Q. Du, Y.-L. Chang, and R. L. King, "High performance computing for hyperspectral remote sensing," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 4, no. 3, pp. 528–544, Jan. 2011.

[28] C. A. Lee, S. D. Gasster, A. Plaza, C.-I. Chang, and B. Huang, "Recent developments in high performance computing for remote sensing: A review," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 4, no. 3, pp. 508–527, Sep. 2011.

[29] J. M. Van Campenhout, "On the peaking of the Hughes mean recognition accuracy: The resolution of an apparent paradox," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-8, no. 5, pp. 390–395, May 1978.

[30] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*. Amsterdam, The Netherlands: Elsevier, 2003.

[31] C. Sima and E. R. Dougherty, "The peaking phenomenon in the presence of feature-selection," *Pattern Recognit. Lett.*, vol. 29, no. 11, pp. 1667–1674, Aug. 2008.

[32] G. Hughes, "On the mean accuracy of statistical pattern recognizers," *IEEE Trans. Inf. Theory*, vol. IT-14, no. 1, pp. 55–63, Jan. 1968.

[33] R. Bellman, *Adaptive Control Processes: A Guided Tour* (Princeton Legacy Library). Princeton, NJ, USA: Princeton Univ. Press, 2015.

[34] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[35] H. Lu and K. Kawaguchi, "Depth creates no bad local minima," 2017, *arXiv:1702.08580*. [Online]. Available: http://arxiv.org/abs/1702.08580

[36] T. Poggio, H. Mhaskar, L. Rosasco, B. Miranda, and Q. Liao, "Why and when can deep-but not shallow-networks avoid the curse of dimensionality: A review," *Int. J. Autom. Comput.*, vol. 14, no. 5, pp. 503–519, Oct. 2017.

[37] M. Raghu, B. Poole, J. Kleinberg, S. Ganguli, and J. S. Dickstein, "On the expressive power of deep neural networks," in *Proc. 34th Int. Conf. Mach. Learn. (JMLR)*, vol. 70, 2017, pp. 2847–2854.

[38] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez, "Convolutional neural networks for large-scale remote-sensing image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 2, pp. 645–657, Feb. 2017.

[39] H. Robinson, A. Rasheed, and O. San, "Dissecting deep neural networks," 2019, *arXiv:1910.03879*. [Online]. Available: http://arxiv.org/abs/1910.03879

[40] X. X. Zhu *et al.*, "Deep learning in remote sensing: A comprehensive review and list of resources," *IEEE Geosci. Remote Sens. Mag.*, vol. 5, no. 4, pp. 8–36, Dec. 2017.

[41] M. E. Paoletti, J. M. Haut, J. Plaza, and A. Plaza, "Deep learning classifiers for hyperspectral imaging: A review," *ISPRS J. Photogramm. Remote Sens.*, vol. 158, pp. 279–317, Dec. 2019.

[42] H. Lee and H. Kwon, "Going deeper with contextual CNN for hyperspectral image classification," *IEEE Trans. Image Process.*, vol. 26, no. 10, pp. 4843–4855, Oct. 2017.

[43] 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 Trans. Geosci. Remote Sens.*, vol. 56, no. 11, pp. 6440–6461, Nov. 2018.

[44] L. Mou, L. Bruzzone, and X. X. Zhu, "Learning features via a recurrent convolutional neural network for change detection in multispectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 2, pp. 924–935, Feb. 2019.

[45] L. Mou and X. X. Zhu, "Learning to pay attention on spectral domain: A spectral attention module-based convolutional network for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 1, pp. 110–122, Jan. 2020.

[46] S. K. Roy, G. Krishna, S. R. Dubey, and B. B. Chaudhuri, "HybridSN: Exploring 3-D–2-D CNN feature hierarchy for hyperspectral image classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 17, no. 2, pp. 277–281, Feb. 2019.

[47] J. M. Haut, M. E. Paoletti, J. Plaza, A. Plaza, and L. Plaza, "Hyperspectral image classification using random occlusion data augmentation," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 11, pp. 1751–1755, Nov. 2019.

[48] M. E. Paoletti, J. M. Haut, J. Plaza, and A. Plaza, "Neighboring region dropout for hyperspectral image classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 17, no. 6, pp. 1032–1036, Jun. 2020.

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

[50] L. He, J. Li, C. Liu, and S. Li, "Recent advances on spectral–spatial hyperspectral image classification: An overview and new guidelines," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 3, pp. 1579–1597, Nov. 2017.

[51] S. Yu, S. Jia, and C. Xu, "Convolutional neural networks for hyperspectral image classification," *Neurocomputing*, vol. 219, pp. 88–98, Jan. 2017.

[52] M. E. Paoletti, J. M. Haut, J. Plaza, and A. Plaza, "A new deep convolutional neural network for fast hyperspectral image classification," *ISPRS J. Photogramm. Remote Sens.*, vol. 145, pp. 120–147, Nov. 2018.

[53] M. E. Paoletti *et al.*, "Capsule networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 4, pp. 2145–2160, 2018.

[54] S. K. Roy, S. Chatterjee, S. Bhattacharyya, B. B. Chaudhuri, and J. Platoš, "Lightweight spectral–spatial squeeze-and-excitation residual bag-of-features learning for hyperspectral classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 8, pp. 5277–5290, Aug. 2020.

[55] K. He and J. Sun, "Convolutional neural networks at constrained time cost," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 5353–5360.

[56] B. Wu *et al.*, "Shift: A zero flop, zero parameter alternative to spatial convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9127–9135.

[57] Z. Zhong, J. Li, Z. Luo, and M. Chapman, "Spectral–spatial residual network for hyperspectral image classification: A 3-D deep learning framework," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 2, pp. 847–858, Oct. 2017.

[58] M. E. Paoletti, J. M. Haut, R. Fernandez-Beltran, J. Plaza, A. J. Plaza, and F. Pla, "Deep pyramidal residual networks for spectral–spatial hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 2, pp. 740–754, Aug. 2018.

[59] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 1492–1500.

[60] X. Kang, B. Zhuo, and P. Duan, "Dual-path network-based hyperspectral image classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 3, pp. 447–451, Mar. 2019.

[61] M. E. Paoletti, J. M. Haut, J. Plaza, and A. Plaza, "Neural ordinary differential equations for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 3, pp. 1718–1734, Mar. 2020.

[62] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, 2010, pp. 807–814.

[63] W. Chen, D. Xie, Y. Zhang, and S. Pu, "All you need is a few shifts: Designing efficient convolutional neural networks for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 7241–7250.

[64] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 1251–1258.

[65] A. G. Howard *et al.*, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*. [Online]. Available: http://arxiv.org/abs/1704.04861

[66] H. Zhong, X. Liu, Y. He, and Y. Ma, "Shift-based primitives for efficient convolutional neural networks," 2018, *arXiv:1809.08458*. [Online]. Available: http://arxiv.org/abs/1809.08458

[67] Y. Jeon and J. Kim, "Constructing fast network through deconstruction of convolution," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 5951–5961.

[68] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*. [Online]. Available: http://arxiv.org/abs/1502.03167

[69] D. Yu, H. Wang, P. Chen, and Z. Wei, "Mixed pooling for convolutional neural networks," in *Proc. Int. Conf. Rough Sets Knowl. Technol.* Cham, Switzerland: Springer, 2014, pp. 364–375.

[70] X. Huang and L. Zhang, "A comparative study of spatial approaches for urban mapping using hyperspectral ROSIS images over Pavia city, Northern Italy," *Int. J. Remote Sens.*, vol. 30, no. 12, pp. 3205–3221, Jun. 2009.

[71] X. Xu, J. Li, and A. Plaza, "Fusion of hyperspectral and LiDAR data using morphological component analysis," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2016, pp. 3575–3578.

[72] S. Babey and C. Anger, "A compact airborne spectrographic imager (CASI)," in *Quantitative Remote Sensing: An Economic Tool for the Nineties*, vol. 1, 1989, pp. 1028–1031.

[73] M. Paoletti, J. Haut, J. Plaza, and A. Plaza, "Deep&dense convolutional neural network for hyperspectral image classification," *Remote Sens.*, vol. 10, no. 9, p. 1454, Sep. 2018, doi: 10.3390/rs10091454.