UNIVERSIDAD DE EXTREMADURA

TESIS DOCTORAL

Procesamiento eficiente y profundo de imágenes hiperespectrales de la observación remota de la Tierra y aplicaciones en tareas de clasificación

MERCEDES EUGENIA PAOLETTI ÁVILA

Programa de doctorado en Tecnologías Informáticas

UNIVERSIDAD DE EXTREMADURA

TESIS DOCTORAL

Procesamiento eficiente y profundo de imágenes hiperespectrales de la observación remota de la Tierra y aplicaciones en tareas de clasificación

MERCEDES EUGENIA PAOLETTI ÁVILA

Programa de doctorado en Tecnologías Informáticas

Con la conformidad:

Dr. Antonio Plaza Miguel *Dr. Javier Plaza Miguel*

May 2020

Acknowledgements

This thesis was supported by Ministerio de Educación through the state grant FPU15/02090 awarded by 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.

Also, this thesis was partially supported by Junta de Extremadura (decreto 14/2018, 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) and by the project "Tools for Mapping Human Exposure to Risky Environmental conditions by means of Ground and Earth Observation Data (EOXPOSURE)", funded by the European Union's Horizon 2020 research and innovation program under the grant agreement no. 734541.

Abstract

Advances in computing technology and remote sensing field have fostered the development of powerful spectrometers that are able to collect large volumes of hyperspectral data. These data are characterized by their high spectral resolution, recording solar radiation and absorption of surface materials by measuring different wavelengths, along the electromagnetic spectrum. As a result, hyperspectral imaging (HSI) is a popular topic within the remote sensing field because of the large and rich amount of both spectral and spatial information it contains, which allows for better characterization and exploitation of the Earth's surface. However, HSI data processing methods must face great challenges, especially those supervised classification methods, due to the high spectral dimensionality of the data (which introduces an important amount of data variability) and the limited availability of training samples to cover all the data (resulting in a poor model fit).

In this context, deep learning (DL) methods arise as an interesting solution to enhance the HSI data processing and classification, reaching promising results in a wide range of applications within computer vision tasks. This thesis focus its efforts in the development of new and efficient DL approaches for HSI classification, processing not only spectral information but also spatial and spectral-spatial features from original HSI data cube and providing more robust solutions to overfitting, high dimensionality, data anomalies and data variability. To illustrate the advantages and benefits of the implemented proposals in comparison with the current state-of-the-art in HSI land cover classification, several experiments have been conducted considering real HSI scenes and performing the corresponding comparison with the available processing methods in the literature.

Table of contents

Compendium of publications

The current thesis is a compendium of the following publications:

1. **M. E. Paoletti**, J. M. Haut, J. Plaza, A. Plaza. Deep learning classifiers for hyperspectral imaging: A review. ISPRS Journal of Photogrammetry and Remote Sensing, vol. 158, pp. 279-317, December 2019. DOI: 10.1016/j.isprsjprs.2019.09.006. [IF(2018)=6.942]. Google Scholar Citations: 13.

2. **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza. Scalable Recurrent Neural Network for Hyperspectral Image Classification. Journal of Supercomputing, accepted for publication, 2020. DOI: 10.1007/s11227-020-03187-0. [IF(2018)=2.157]. Google Scholar Citations: 1.

3. **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, vol. 145, Part A, pp. 120-147, November 2018. DOI: 10.1016/j.isprsjprs.2017.11.021. [IF(2018)=6.942]. Google Scholar Citations: 146.

4. **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza. Neighboring Region Dropout for Hyperspectral Image Classification. IEEE Geoscience and Remote Sensing Letters, accepted for publication, 2020. DOI: 10.1109/LGRS.2019.2940467. [IF(2018)=3.534]. Google Scholar Citations: 0.

5. J. M. Haut, **M. E. Paoletti**, J. Plaza, A. Plaza and J. Li. Visual Attention-Driven Hyperspectral Image Classification. IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 10, pp. 8065-8080, October 2019. DOI: 10.1109/TGRS.2019.2918080. [IF(2018)=5.630]. Google Scholar Citations: 10.

6. **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza. Neural Ordinary Differential Equations for Hyperspectral Image Classification. IEEE Transactions on Geoscience and Remote Sensing, vol. 58, no. 3, pp. 1718-1734, March 2020.
DOI: 10.1109/TGRS.2019.2948031. [IF(2018)=5.630]. Google Scholar Citations: 0.

7. **M. E. Paoletti**, J. M. Haut, R. Fernandez-Beltran, J. Plaza, A. Plaza, J. Li and F. Pla. Capsule Networks for Hyperspectral Image Classification. IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 4, pp. 2145-2160, April 2019. DOI: 10.1109/TGRS.2018.2871782. [IF(2018)=5.630]. Google Scholar Citations: 51.

General overview of the Thesis

1.1 Challenges of hyperspectral data classification

Advances in communication and computer technologies have provided a great development in remote sensing and Earth Observation fields, allowing the acquisition of high quality images of the Earth's surface, through remote sensors located on aerial and satellite platforms, which provides a large amount of data with different spatial, spectral and temporal resolutions. The analysis and processing of these remote data have proven to be very useful in a wide range of social-economic activities, such as natural resource management, planning of urban areas, management of agricultural fields, risk prevention and target detection. In fact, remote sensing missions are currently developed and launched with powerful sensors that are able to perform a detailed characterization of the observed surface by gathering the acquired data into images with:

- a high spatial resolution, understood as the level of detail of an image perceived by the human eye, and expressed in meters per pixel (mpp),
- a high spectral resolution, understood as the number of measurements in the spectral domain over different wavelengths, usually from the shortwave infrared (SWIR) and near infrared (NIR) to the visible spectrum, and expressed as the number of spectral bands contained in the data cube,
- and a wide range of temporal resolution, understood as the availability of data from the same observation area at different time points, closely linked to the revisit times of the sensors.

This thesis document focuses particularly on those remote sensing products and images of the second type, which are characterized by their extensive information contained in the spectral domain, the so-called hyperspectral imaging (HSI). Specifically, imaging spectrometers generates HSI products as large data cubes, where each pixel captures the solar radiation reflected by the scene materials in hundreds of narrow bands along the electromagnetic

spectrum, providing a "continuous" spectral signature of the observed materials rather than the discrete information of RGB color models and multispectral images (MSI). This spectral signature can be interpreted as the unique and distinctive fingerprint of each terrestrial material, allowing us to "see what the human eye is not able to see". Such wealth of spectral information is very useful to make a detailed study of the observed surface, not only to check its composition but also the state of the soil. Through the application of pattern recognition and machine learning techniques, HSI images have revolutionized the field of applied remote sensing, being widely used in precision agriculture activities, the study of pests and crops diseases, the analysis of the water reserves quality, the chemical composition of the soil, soil erosion and degradation of coastal zones, for instance.

Multiple methods have been developed with the goal of analyzing and processing effectively, efficiently and with a reasonable computation time, the information contained in the HSI data cubes, exploiting both the spectral and spatial information. Some of these methods can be categorized in different groups, depending on the functionality of each one, such as methods to enhance the spatial resolution (pan-sharpening, for instance), denoising and recovering techniques to enhance the spectral information of the image, spectral unmixing techniques, spectral dimensionality reduction and band selection methods to reduce spectral redundancies and correlations, anomalies and change detection approaches and techniques to categorize and classify the land cover captured in each pixel, among others. In particular, this thesis document emphasizes those are methods based on land cover classification, whose purpose is to assign to each spectral pixel that composes the scene, the category or label corresponding to the material collected at that pixel.

Framed within machine learning field, the scientific community has developed a multitude of spectral, spatial and spectral-spatial classifiers to perform HSI land cover classification, which are often inspired by widely used pattern recognition algorithms (such as the k-means or the support vector machine, among others) and are specially adapted to process the spectral-spatial information contained into the HSI data cube. However, the classification of HSI data is an arduous problem due to the high dimensionality of HSI data, which leads to a number of difficulties that hinder the performance of traditional classifiers. In a more detailed way, the main challenges of the HSI data classification are related to the following factors:

- Although each spectral pixel contained into the HSI scene provides a great deal of information in the form of spectral signature, its high dimensionality increases the complexity of classification models. This is due to the so-called *peaking paradox*, which demonstrates that the use of additional features increases the number of statistical parameters employed by the models to define each land cover class.

- Due to the higher number of statistical parameters that must be estimated, classification models need more training data with the aim of being able to adjust them correctly. This may give rise to the *curse of dimensionality*, whereby, beyond a certain spectral size, the training set does not contain enough information to correctly tune the large amount of parameters. As a result, the models are poorly adjusted to the data, producing over-fitting problems (*Hughes phenomenon*).

- In addition to the behaviour of classifiers when faced with large data, the internal characteristics of these data must be taken into account, too. In this sense, the spectral information contained into HSI data cubes can be affected by several artifacts that make the classification process harder. On the one hand, uncontrolled changes in the light captured by the remote sensor can introduce a great variability within the data, for instance different variations in scene illumination or changes in atmospheric conditions. On the other hand, some anomalies and limitations in the sensor can hinder the quality of the spectral information. It must be noted that, in a simple way, the spectrometer measures the radiation by scattering the light beam that passes through an entrance slit by means of a refracting element. In this sense, the width of the slit can introduce some blurring and smoothing effects that hinders the spectral resolution. Also, other factors, such as a poor radiometric accuracy and signal-to-noise ratio (SNR), can impact negatively on the quality of the captured data.

- Furthermore, keeping a low bandwidth in order to record the spectral information in hundreds of continuous narrow bands involves a compromise between the spectral and the spatial resolutions due to technological limitations. In this sense, the lower the bandwidth, the lower the received radiation signal, reducing the spatial resolution. As a result, medium or low-spatial resolution HSI cube are obtained with high mpp, where each pixel represents a large area from the target surface, so its spectral signatures are very mixed. This increases the variability of samples that belongs to the same land cover, while intensify the similarity between pixels belonging to edge zones.

- Finally, the acquisition of labeled HSI samples is already very expensive. This is due to several factors: on the one hand, satellite HSI-based observation missions remain poorly represented within Earth Observation field due to their technical constraints and practical challenges. As a result, the number of current operational spectrometers are still low compared to other types of remote sensors, such as those spatial-high-resolution and MSI devices in Landsat, Sentinel and SPOT systems. On the other hand, the airborne spectrometers cover much smaller areas than the satellite-based sensors, so the amount of HSI data sets is limited. In addition, repositories with labeled HSI

scenes are usually not publicly available, being the tagging of each pixel and arduous and expensive task.

1.2 Efficient deep learning methods for effective classification of hyperspectral remote sensing data

This thesis focuses its efforts on solving these limitations by developing powerful HSI classifiers that are highly effective and computationally efficient. In this sense, deep learning techniques are particularly interesting for this thesis, due to their great success within the field of computer vision, which has led them to stand out as the current state of the art. Moreover, deep learning methods offer a great versatility, exhibiting a large generalization power that makes them universal function approximators. In this sense, deep learning architectures can model any kind of optimization problem as a mapping function with some inputs and the desired output, where the mapping function is usually implemented as a deep stack of operational layers. This hierarchical-based architecture offers a high flexibility when designing the mapping function, allowing the use of different types of layers and connections. Moreover, deep models do not require any prior data or handmade-extracted information about the data, while the forward-backward step mechanism allows to adjust them to the data automatically. Despite these attractive properties, deep learning methods also exhibit several limitations when classifying remote sensing HSI data, such as their propensity for over-fitting when there are few training samples available, the high computational burden due to the large amount of parameters that must be adjusted and the training degradation of very deep networks, among others. In order to overcome these shortcomings, this thesis proposes the study of new deep learning architectures to perform the classification of remote sensing HSI data in an effective and efficient way, applying techniques of high performance computing (HPC) (in particular the parallelization of the models over graphics processing unit -GPU- devices) to improve not only the quality of the classification results but also to develop an efficient management of deep models in terms of memory and computational resources consumption.

More specifically, this thesis has the following scientific objectives:

1. The first objective is to analyze and identify the current state-of-the-art in the analysis and processing of HSI data from remote Earth observation, exploring in the available literature those DL techniques that have been successfully employed for land cover classification.

2. The second objective is to explore in detail the problem of hyperspectral analysis, studying the characteristics of current operational sensors and the available data repositories, identifying the main characteristics, challenges, difficulties and limitations of the available data.

3. The third objective is to analyze different deep learning based classification methods, observing their strengths and weaknesses in order to propose, implement and validate methodological and computational improvements of these methods to enable faster, more accurate and more efficient analysis of the spectral-spatial information contained in the captured hyperspectral images, by:

    - improving deep learning architectures and algorithms, and
    - developing distributed many-core solutions based on GPUs, characterized by their high performance and low power consumption comparison with conventional multicore CPUs,

    and conducting an exhaustive comparative analysis of the improvements developed in real applications, using real hyperspectral images obtained by different current spectrometers, such as AVIRIS and ROSIS instruments in order to validate them.

4. The last objective is to validate the new developed algorithms and models for the classification of the hyperspectral images, evaluating with special attention those improvements obtained both in terms of accuracy (through well-known metrics such as overall accuracy -OA-, average accuray -AA- and kappa coefficient -K-) as well as in terms of complexity and execution time, through a comprehensive and fair comparison with the classification methods employed by the scientific community.

Following these objectives, this thesis by compendium presents the results obtained throughout the PhD period in six different publications, which have been submitted and published by several international journals with impact index:

- The first article, entitled *Scalable Recurrent Neural Network for Hyperspectral Image Classification* [249], introduces one of the most widely used deep learning architectures in the field of computer vision, the recurrent neural network (RNN), which have been adapted to perform the spectral-based classification of HSI scenes. In particular, the new simple recurrent unit (SRU) is tested, applying a GPU-parallelized implementation in order to obtain not only good accuracy results but to achieve also a competitive performance in terms of run times and data scalability. The quantitative and qualitative results obtained from the experiments demonstrate the benefits of the proposal, whose computational results stand out in comparison with current RNN implementations.

- The second article, entitled *A New Deep Convolutional Neural Network for Fast Hyperspectral Image Classification* [246], takes a further step into the deep learning field, applying the most popular architecture, the convolutional neural network (CNN), to the classification of HSI scenes employing spectral-spatial information in an efficient way. In addition to conducting a detailed study of how spatial information affects the classifier, the results obtained during the experimentation demonstrate the superiority of the proposal when compared to other implementations.

- The third article, entitled *Neighboring Region Dropout for Hyperspectral Image Classification* [247] delves into the performance of the CNN model. In particular, this paper focuses on those mechanisms that improve the generalization and learning procedure of the network based on data occlusion and dropout techniques. In this context, it proposes a new spatial occlusion method to enhance the performance of the model during the training stage when dealing with HSI data classification problems. The proposal achieves outstanding results in comparison with standard regularization methods.

- The fourth article, entitled *Visual Attention-Driven Hyperspectral Image Classification* [138], follows the research line of the previous ones, studying new implementations to improve and enhance the CNN model accuracy and performance during HSI classification tasks. In this case, the proposal takes advantage of residual and skip connections along the CNN model, applying visual attention mechanism to enforce the most interesting spectral-spatial features contained into the HSI scene by applying masks automatically learned by the model. Obtained results demonstrate that the approach is more robust to over-fitting than traditional implementations, achieving higher accuracy values with very small training sets.

- The fifth article, entitled *Neural Ordinary Differential Equations for Hyperspectral Image Classification* [248], proposes a re-interpretation of the residual architecture for CNN models. In this context, residual network (ResNet) for HSI data classification is able to reach very good results in terms of accuracy in comparison with standard CNNs, however its complexity and depth drastically increase the memory and runtime consumption as we add residual blocks. To overcome this limitation, the proposal re-interpret the traditional-discrete ResNet as a continuous-time ordinary differential equation (ODE), drastically reducing the number of parameters (and therefore the memory consumption) to be trained, while maintaining or even improving the accuracy results compared to the current state-of-the-art in HSI data classification.

- Finally, the sixth article, entitled *Capsule Networks for Hyperspectral Image Classification* [244], redesigns the CNN model, grouping the basic convolution layer into capsules of different levels, which are connected through a dynamic routing that allows the learning of spatial relationships, something that CNN cannot achieve due to mechanisms such as pooling. The result is a model which is significantly more robust to over-fitting problems when training with few training samples than standard CNNs, overcoming the results obtained by the current state-of the-art in HSI classifiers.

All the proposed models have been implemented on GPUs, seeking to optimize computational resources. Moreover, as we can observe, the considered publications in this document maintain the same research line, which is mainly focused on the classification of the land cover of HSI scenes through the implementation of new and efficient deep learning techniques, which are adapted to the computational power of current computing devices to provide accurate results in reasonable execution times. Furthermore, these techniques aim to overcome the limitations of traditional classification algorithms when dealing with complex HSI data, extracting spectral, spatial and spectral-spatial features from the HSI data cube and providing more robust solutions to over-fitting, high dimensionality, anomalies and variability of HSI data. To illustrate the advantages and benefits of the implemented proposals in comparison with the current state-of-the-art, the conducted experiments consider a large variety of real HSI scenes, which have been widely used by the scientific community as benchmarks.

In order to introduce the above mentioned scientific contributions in a clear and contextualized way, this thesis will be organized following the structure proposed by the survey conducted by the PhD student entitled *Deep learning classifiers for hyperspectral imaging: A review* [243], which presents a systematic review about the available deep learning architectures and their application for HSI data processing in classification tasks, highlighting their strengths and weaknesses in comparison with shallow models and standard machine learning methods, and analyzing the current trends. In this context, the original paper have been also included into this compendium of publications as scientific review.

Abbreviations

List of acronyms and abbreviations used in this thesis:

AA	Average accuracy
AE	Autoencoder
AIS	Airborne imaging spectrometer
ANN	Artificial neural network
AVIRIS	Airborne visible infrared imaging spectrometer
AVIRIS-NG	Airborne visible infrared imaging spectrometer next generation
BIL	Band-interleaved-by-line
BIP	Band-interleaved-by-pixel
BIP	Big Indian pines scene
BSQ	Band sequential
CapsNet	Capsule network
CASI	Compact airborne spectrographic imager
CHRIS	Compact high resolution imaging spectrometer
CNN	Convolutional neural network
CONV	Convolution layer
CPU	Central processing unit
DAE	Deep autoencoder (synonym of SAE)
DBN	Deep belief network
DenseNet	Dense network
DESIS	DLR Earth sensing imaging spectrometer
DL	Deep learning
DN	Digital number
DNN	Deep neural network
EDNN	Extremely deep neural network
ELM	Extreme learning machine
EnMAP	Environmental mapping and analysis program
EO	Earth Observation

ER	Evidential reasoning
FC	Fully connected
FE	Feature extraction
FOV	Field of view
GAN	Generative adversarial network
GLAC	Gradient local auto-correlation
GPU	Graphics processing unit
GRU	Gated recurrent unit
GSD	Ground sample distance
HISUI	Hyperspectral imager suite
HOG	Histogram of oriented gradients
HPC	High performance computing
HSI	Hyperspectral imaging
HYDICE	Hyperspectral digital imagery collection experiment
HYMAP	Hyperspectral mapper
HyspIRI	Hyperspectral infrared imager
IAP	invariant attribute profile
IFOV	Instantaneous field of view
IP	Indian pines scene
IS	Imaging spectrometry
ISODATA	Iterative self-organizing data analysis technique
K	Kappa coefficient
KNN	K-nearest neighbors
KSC	Kennedy space center scene
LBP	Local binary patterns
LDA	Linear discriminant analysis
LR	Logistic regression
LReLU	Leaky rectified linear activation function
LSTM	Long short term memory
MFN	Multilayer feedforward network
ML	Machine learning
ML	Maximum likelihood
MLP	Multilayer perceptron
MLR	Multinomial logistic regression
MP	Morphological profiles
mpp	Meters per pixel

MSE	Mean squared error
MSI	Multispectral imaging
NIR	Near infrared
OA	Overall accuracy
ODE	Ordinary differential equation
PCA	Principal component analysis
POOL	Pooling layer
PR	Pattern recognition
PReLU	Parametric rectified linear activation function
PRetanh	parametric rectified tanh
PRISM	Portable remote imaging spectrometer
PRISMA	Precursore iperspettrale della missione applicativa
RBF	Radial basis function
RBM	Restricted Boltzmann machine (RBM)
ReLU	Rectified linear activation function
ResNet	Residual neural network
RF	Random forest
RNN	Recurrent neural network
ROSIS	Reflective optics system imaging spectrometer
RS	Remote sensing
SAE	Stacked autoencoder
SAM	Spectral angle mapper
SC	Sparse coding
SDAE	Stacked denoising autoencoder
SDAE	Spatial updated deep autoencoder
SeLU	Scaled exponential linear unit
SHALOM	Spaceborne hyperspectral applicative land and ocean mission
SIFT	Scale invariant feature transform
SL	Statistical learning
SNR	Signal-to-noise ratio
SOM	Self-organizing map
SRU	Simple recurrent unit
S-SAE	Segmented stacked autoencoder
SURF	Speeded-up robust features
SV	Salinas valley scene
SVM	Support vector machine

SWIR	Short-wave infrared
UAV	Unmanned aerial vehicle
UH	University of Houston scene
UP	University of Pavia scene
UV radiation	Ultra-violet radiation
VDNN	Very deep neural network
VSWIR	visible-to-shortwave infrared
VRU	Vanilla recurrent unit
WOS	Web of Science

Hyperspectral imaging classification

3.1 Introduction: Remote sensing and imaging spectrometry

Within the field of Earth Observation (EO), the terms *remote sensing* (RS) and *imaging spectrometry* (IS) are closely linked, where the former emphasizes the recording of information through non-contact observation procedures, i.e. without the observed target and the capture device coming into direct physical contact [102, 299, 330], and the latter emphasizes the nature of the gathered information. Focusing on RS, despite there is no single and universally accepted interpretation about RS [111], some authors have refined the above broad definition, focusing it on the capture, treatment, analysis and interpretation of some type of emitted or reflected signal from the Earth's surface by measuring the behaviour of the signal waves when they interact with terrestrial matter through remote sensors located on mobile platforms, such as spacecraft and aircraft [44, 315]. This interpretation leaves us a very wide margin of study, embracing remotely sensed data based on the measurement of a wide variety of signals, such as seismic, sonic and electromagnetic waves. In this context, this thesis focuses its study on the RS dedicated to the analysis of the electromagnetic radiation reflected by terrestrial materials.

Electromagnetic radiation can be understood as a wave motion, for which electric and magnetic fields oscillate periodically and perpendicular to each other and to the direction of propagation. In particular, it spreads in harmonic and sinusoidal fashion defined by its wavelength λ, amplitude A and frequency ν, propagating at a speed c, which can reach 3×10^8 meters per second (m/s) the in the vacuum space. These parameters are closely related through Eq. (3.1):

$$c = \nu\lambda, \quad \lambda = c/\nu \quad \text{and} \quad \nu = c/\lambda \tag{3.1}$$

Moreover, the frequency ν and wavelength λ can be determined by the energy of a quantum, which is the minimum unit and fundamental particle that carries the electromagnetic energy

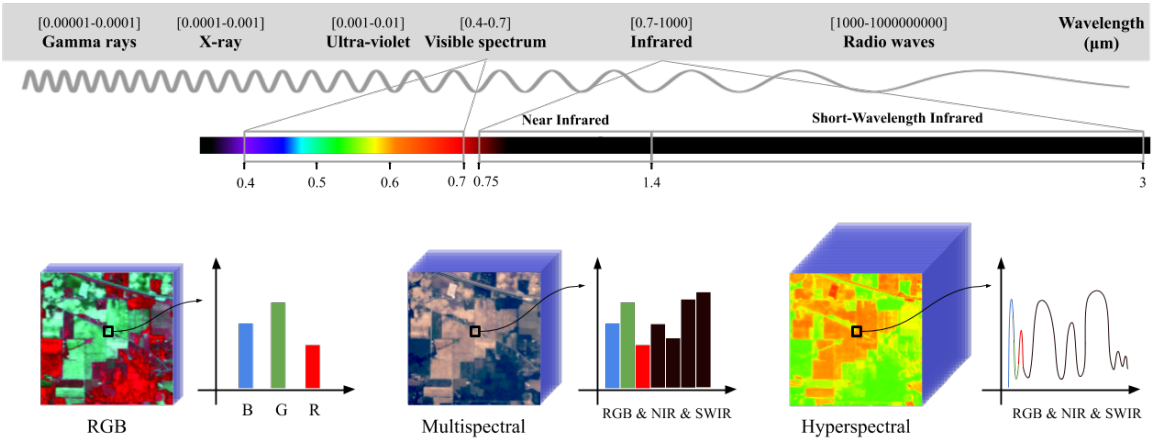


Fig. 3.1 Detail of the electromagnetic spectrum, coupled with traditional remotely sensed images according to their spectral resolution

[5, 274]. In this sense, the greater the energy of the quantum (denoted as Q), the greater the frequency ν. On the contrary, the greater the energy of the quantum, the smaller wavelength λ. This relationship can be extracted from Eq. (3.2):

$$Q = hc/\lambda = h\nu \tag{3.2}$$

where h denotes the Planck's constant. As a result, the electromagnetic spectrum can be obtained as the set of all electromagnetic radiations, arranged according to their wavelength/frequencies in different regions of the spectrum, which in turn defines their behaviour during the emission, transmission and absorption of the energy they convey. Fig. 3.1 displays the electromagnetic spectrum arranged from the shortest wavelengths (i.e. the waves with highest frequencies) to the longest wavelengths (and therefore, the waves with lowest frequencies).

Nowadays, all wavelengths are used for different tasks and applications, from the application of gamma rays for the treatment of cancers through radiotherapy procedures, to the use of radio waves for the transmission of information such as radio and television broadcasts or WiFi signals. However, in RS field the use of certain wavelengths is limited by the atmosphere absorption phenomena [276]. For instance, the ultra-violet radiation (or UV, which is in the range 0.001-0.01μm) is largely scattered by the atmosphere, so it is usually discarded in RS applications. On the contrary, the visible and infrared regions (in the ranges 0.4-0.75μm and 0.75-1000μm, respectively) contain a significant number of wavelengths that are unaffected by atmospheric absorption, identified as *atmospheric windows* [95, 296], while the radio waves are almost unaffected, being able to go through the clouds. In this way,

the following spectral regions have been identified as the most interesting for the exploitation of RS information [276]: i) the so-called optical wavelengths of the visible and infrared regions, ii) the thermal wavelengths of the infrared region, and iii) the radio wavelengths of the microwave and radio spectral regions. Taking this into account, the field of study of this thesis can be defined more closely, being mainly focused on the analysis of optical RS in general, and on the solar radiation reflected by surface's objects and material in particular.

At this point, it is interesting to introduce the IS as a technique to analyze the physical-chemical characteristics of a target by measuring its reflected, emitted and transmitted radiation along the electromagnetic spectrum [120], obtaining much more data than the human eye is able to detect. In this regard, the use of imaging techniques has become a key method within RS field, providing a huge amount of data collected from one surface area by measuring over a wide range of spectral wavelengths. The combination of both techniques has boosted the electro-optical RS field [13], developing a wide variety of activities, mechanisms and technologies for the accurate acquisition of electromagnetic information about the Earth's surface, as well as powerful sensors that are able to collect valuable data about the target materials and objects that other type of sensors cannot cover.

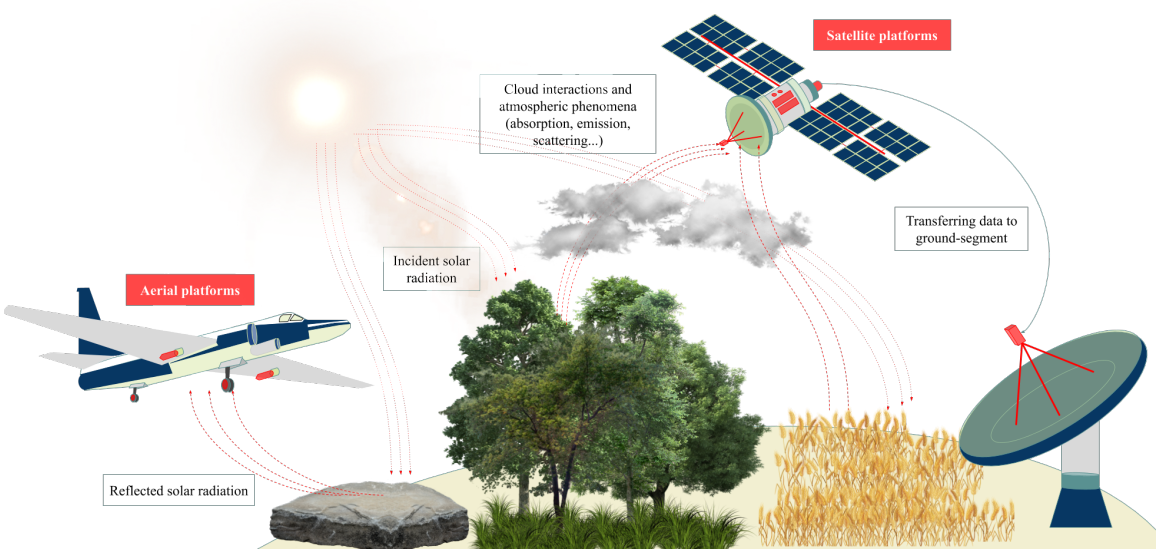


Fig. 3.2 Remote sensing system overview based on capturing solar radiation reflected by materials on the Earth's surface. Imaging spectrometers capture in each element of the scene the degree of absorption/reflection of the signal along different wavelengths within the electromagnetic spectrum. The raw data obtained is usually transferred to the ground segment for further processing.

In particular, great efforts have been invested in those *visible-to-shortwave infrared* (VSWIR) specialized imaging spectrometers, which are able to measure the spectrum in

the ranges from visible (in the range 0.4-0.75μm) to near-infrared (the so-called NIR, in the range 0.75-1.4 μm) and shortwave infrared (also known as SWIR, in the range 1.4-3μm) wavelengths, sponsored by the significant advances in hardware devices and software frameworks [54, 195]. This has led to a great development of the the remote sensing hyperspectral imaging (HSI) techniques, which has gained increasing popularity within the EO field [53]. Looking at Fig. 3.1, we can observe that HSI systems collect the spectral information into hundreds of narrow and continuous bands, where each band contains the reflectance measurement at the corresponding wavelength [120], denoted as $\rho(\lambda)$:

$$\rho(\lambda) = (E_R(\lambda)/E_I(\lambda)) \cdot 100 \tag{3.3}$$

where $E_R(\lambda)$ is the energy at wavelength λ reflected from the captured material and $E_I(\lambda) = E_R(\lambda) + E_A(\lambda) + E_T(\lambda)$ is the original energy at wavelength λ that impacts the observed material, which can be obtained as the combination of those energies reflected, absorbed $E_A(\lambda)$ and transmitted $E_T(\lambda)$ by the material. As a result, a distinctive, unique and continuous spectral signature is obtained for each captured target, as opposed to broad-band systems, such as standard RGB color maps and multispectral sensors (MSIs), which discretize the spectral information into a few (at most tens) spectral bands, under-sampling the available information [320]. In this sense, HSI sensors gather extremely valuable and rich information to understand the different phenomena of the Earth's surface on a local and global scale through the availability of HSI sensors located on a wide range of platforms, including manned and unmanned aerial platforms, satellite platforms and even in-situ ground imaging platforms [4, 21, 239].

Related to this, Fig. 3.2 provides a general overview of a standard HSI-RS system, while Fig. 3.3 shows some details of a standard HSI sensor (particularly, a push-broom HSI sensor). In this regard, one of the most widely used methods for HSI data acquisition is to raster-scan the scene as the aerial/satellite platform moves along the target region's surface (the so-called along-track direction), where HSI sensors works as line scan spectrometers [222, 293, 316]. By combining spectroscopy and digital imaging, the sensor device images a footprint or swath (which depends on the instantaneous field of view, IFOV) of the target area, acquiring a sequence of pixels and producing at an integration time a band-interleaved-by-line (BIL) matrix with the reflectance measurements along different wavelengths for each pixel. To do this, the sensor projects the scanned line onto an entrance slit and then refracts the signal at hundreds of different wavelengths (considering a narrow bandwidth, which is usually between 4-15nm) through a dispersing element. This element changes the speed of the electromagnetic radiation, bending different wavelengths at different angles as they slow down at different speeds. The final sensing elements collects the information contained into

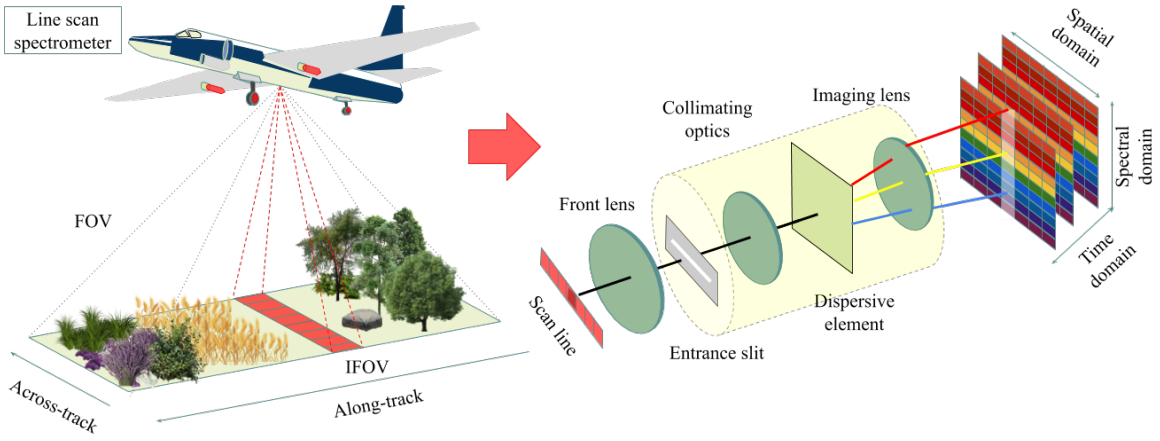


Fig. 3.3 Standard scheme of an imaging spectrometer following spatial scanning procedure. As the platform moves, the sensor captures the radiation reflected by the materials within its field of view (FOV), scanning a swath of the target area depicted by the instantaneous field of view (IFOV). This radiation is treated through a lens system and refracted at different wavelengths by a dispersing element. Then, the refracted signal is recorded by the camera's sensor at every spatial position.

these wavelengths, creating the matrix by assigning a digital number (DN) to each pixel. As the mobile platform moves across the observed region (it must be synchronized with the capture-rate of the sensor), multiple matrices are gathered and staked together in a BIL encoded data cube [10, 105]. After some corrections, the raw HSI data cube is transmitted to the ground segment, where its final post-processing is made[1]. As a result, the final HSI data cube is obtained, which can be interpreted as a batch of several images, where each one correspond to a spectral band with wavelength λ.

From its early applications in the 1980s as a pure research tool [119, 271], where HSI sensors located on aerial platforms recorded the surface information in different flight campaigns [28, 184, 78], such as the Jet Propulsion Laboratory instruments Airborne Imaging

[1] In addition to line scanning performed by push-broom imaging spectrometers, there are other techniques for HSI data acquisition, such as point scanning, which is performed by whisk-broom imaging spectrometers by replacing the entrance slit with an aperture point. These sensors acquire all the spectral bands pixel by pixel, imposing that sensors have to move not only along-track but also across-track directions. The resulting data cube is stored following the band-interleaved-by-pixel (BIP) encoding. Comparing line scanning with point scanning, some works point out the simpler operation and higher signal-to-noise ratio (SNR) of push-broom sensors, coupled with their more compact and less heavy design in comparison with whisk-broom devices [4, 76, 105]. However, push-broom sensors suffer some artifacts that jeopardize the spectral information [230].

Spectral, wavelength or plane scanning is also another interesting HSI acquisition technique, where stationary sensors employ band-pass filters to spectrally scan the entire scene band by band, alternating the filters to capture the desired wavelength. As a result, the data cube obtained is stored following a band sequential (BSQ) strategy [4]. More recently, some HSI data acquisition approaches explore the non-scanning or snapshot imaging techniques [129], which are able to collect the spectral-spatial features of the scene during a single detector integration period.

Spectrometer (AIS) [323] and the Airborne Visible/InfraRed Imaging Spectrometer (AVIRIS) [124], the VSWIR-IS has evolved not only for research purposes but also for operational and commercial applications [100, 301], developing a wide range of aerial and satellite imaging spectrometers, and recently also stationary and hand-held sensors that are currently producing high-quality HSI data [79, 117, 212]. For instance, Table 3.2 provides some details of the most popular spectrometers [268] (it should be noted that not all sensors are currently operational, for instance Hyperion sensor was decommissioned on March 20, 2017 [165], while the updated HyspIRI mission is scheduled for release in 2020/2021 [159]).

Table 3.2 Some popular airborne and satellite HSI sensors/missions. Some spectral-spatial features are provided. On the one hand, focusing on spectral characteristics, this table provide the number of collected bands, the range of considered wavelength (in μm) and the bandwidth (in nm). On the other hand, the ground sample distance (GSD, in mpp) is supplied as the main spatial feature.

	Sensor	Bands	Range	Width	GSD
Airborne	AVIRIS [124]	224	0.36-2.45	10	20
	AVIRIS-NG [42]	600	0.38-2.51	5	0.3-4.0
	CASI [16]	144	0.36-1.05	2.4	2.5
	HYDICE [278]	210	0.40-2.50	10.2	1-7
	HYMAP [73]	126	0.45-2.50	15	5
	PRISM [231]	248	0.35-1.05	3.5	2.5
	ROSIS [186]	115	0.43-0.86	4	1.3
Satellite	CHRIS [80]	62	0.415-1.05	5-15	18-36
	DESIS [92]	180	0.40-1.00	3.30	30
	EnMAP [126]	228	0.42–2.40	5.25-12.5	30
	HISUI [162]	185	0.4-2.5	10-12.5	30
	Hyperion [255]	220	0.40-2.50	10	30
	HyspIRI [196]	-	0.38-2.51	10	30
	PRISMA [261]	237	0.40-2.50	$\leq$12	30
	SHALOM [100]	241	0.40-2.50	10	10

Some of these sensors exhibit impressive data acquisition rates, which are usually in the order of gigabytes per day (GB/day). In particular those spectrometers located at spacecraft platforms are able to acquire huge volumes of HSI data, such as Hyperion, which was able to record 1.6 TB/day. Nowadays, the HISUI instrument is currently collecting 690 GB of HSI data per day, and the SHALOM device has an acquisition rate of 837 GB/day, while EnMap and PRISMA missions are capturing 75 GB/day and 51.25 GB/day respectively [267]. Considering airborne spectrometers, they are limited by both the maximum flight time of the airborne platform and the available storage capacity. For instance the whisk-broom AVIRIS sensor is capable of acquiring 9 GB/h, with a data capacity of 10 GB per flight [318], while its airborne platform has a maximum flight time of 5-6.5 hours.

The large flow of HSI data generated coupled with the impressive amount of both spatial and spectral information they provide, have made HSI data a powerful source of RS information, allowing the accurate land cover categorization at pixel level. In this sense, the exploitation of this kind of data is reaching promising results in a wide range of applications [155, 174, 320, 320]. The vast majority of applications are focused on the management, exploitation and conservation of natural resources[2], followed by agriculture activities[3] [4, 232] and urban planning [145, 146, 213]. The potential of HSI data in the prevention and management of natural risks and disasters has also been demonstrated [281, 320, 326]. Finally, other activities in which it is possible to exploit the spectrum-spatial information of HSI cubes are military and defense applications [40, 122, 277] and archaeological prospects [49, 291], being the NIR and SWIR information pretty useful for buried structures detection.

As a result of its application in this wide range of social and economic activities, the demand for HSI processing methods has increased over the past few years, positioning HSI's technology and data products as a hot topic within RS [243]. In this sense, it is mandatory to design, develop and implement effective techniques and methodologies to efficiently extract the information contained in the HSI cube. There are currently a large variety of HSI processing methods, which can be classified into the following categories, depending on their purpose [50]:

1. HSI data restoration [112, 358] and denoising [59, 217, 335, 339]: the purpose of these HSI processing methods is to reduce those anomalies and noises introduced both by the sensor's own electronic, optical and thermal artifacts and by atmospheric phenomena.

2. Resolution enhancement [93, 228, 348, 347]: usually, HSI scenes exhibit a low spatial resolution because of the technical limitations of the sensor, whereby the narrower the bandwidth, the worse the spatial resolution. The goal of these methods is to enhance the spatial resolution of HSI data, for instance by combining the spatial information of

[2]For instance, in forestry and natural environment management, some relevant works focus their research on exploit the spectral-spatial information contained into HSI data cubes to analyze the status and health of forests [75, 177, 298] and those areas of special environmental interest, such as mangrove forests [148], wetlands [3], coastal zones [258, 295] and other ecosystems [125]. Other works focus on the application of HSI techniques for the management of marine and water resources, by analyzing the water quality in lakes, rivers and coastal zones [38, 39, 94, 179, 240], for instance. Also, some research have successfully employed HSI data to map mineral deposits [2, 275, 292] and to analyze soil degradation [373].

[3]There are some interesting works that explore the potential of HSI data in precision agriculture [128], for instance by studying the water status of crops for irrigation scheduling [283] or even the crop water content [52] and chemical features [118, 227, 314]. Other works propose the analysis and control of crop variability through HSI data [343]. Also, some research proposes the use of HSI data as an effective tool for pest and crop diseases control [257, 306, 365], while others consider this kind of data for analyzing the effect of agricultural activities on soil erosion and degradation, and the generation of associated residues [11, 20, 51].

high-resolution scenes with the spectral information of the HSI cube or by applying super-resolution techniques.

3. Dimensionality reduction [41, 137, 297] and band selection [89, 127, 156, 289]: due to the dense and narrow spectral sampling of HSI instruments, the resulting data cubes if affected by a high correlation between consecutive bands, so there is a high degree of spectral redundancy. The purpose of these HSI processing methods is to reduce the high spectral dimensionality of HSI data cubes by removing the redundant information and keeping the most significant and discriminative information.

4. Data compression [70, 259, 266]: on the one hand, the huge amount of spectral data increases dramatically the storage and computing requirements, making HSI data cubes difficult to handle. On the other hand, standard compression algorithms become obsolete due to the different behaviors and relationships between the spectral and spatial domains. The purpose of these methods is to compress the 3-dimensional cube while keeping constant spectral-spatial relationships of the data cube.

5. Spectral unmixing [33, 147, 149, 236, 288, 300, 369]: due to the trade-off between the bandwidth and the spatial resolution of the HSI data cube, it is quite common for HSI pixels to cover large areas of the target surface, capturing different types of materials and objects. As a result, the obtained spectral signatures are very mixed. The main purpose of these methods can be divided into two goals. On the one hand, to estimate the number of those pure elements of the scene whose spectral signature is not mixed (the so-called endmembers). On the other hand, to estimate the abundances of these end members in each pixel vector of the HSI cube.

6. Image segmentation [199–201]: the purpose of these methods is to partition the HSI data cube into spatially-connected and non-overlapped regions, following a homogeneity criterion (which can be based on spectral and/or contextual information).

7. Target [87, 220, 363], object [322, 364] and anomaly [170, 312, 342] detection: the main purpose of these methods is to detect a particular material within the HSI scene, by separating it from the rest of the background.

8. Data classification [46, 99, 116, 243]: the goal of these methods is to assign a label to each spectral pixel contained into a HSI data cube. These labels represents different land-cover materials, such as man-made materials, crops, artificial structures, different soil types... As a result a classification map is obtained. The goal of these methods is

to analyze those spectral, spatial and spectral-spatial features that are characteristic of each class.

These techniques are not mutually exclusive; on the contrary, they can be combined to improve the extraction, study and application of spectral and also spatial information contained in the HSI scenes. In particular, this thesis focuses its efforts on the classification of HSI data because of their great potential in the study and analysis of land cover and land use [65, 374], being one of the most widespread techniques for HSI data exploitation within the RS community [55]. In this sense, the following section will delve into the HSI data classification problem.

3.2 Hyperspectral data classification

Before going deeper into the classification of HSI data, a few mathematical notations should be introduced. As discussed above, HSI data is a collection of several hundred images captured from the same scene, where each one contains the corresponding reflectance measurements obtained at the respective wavelength λ. In this sense, we can observe the HSI scene as the 3-dimensional array $\mathbf{X} \in \mathbb{N}^{n_1 \times n_2 \times n_b}$, where $n_1 \times n_2$ indicates the height and width spatial dimensions and n_b the number of channels/bands at the spectral dimension. Therefore, each band contains $n_1 \times n_2$ pixels, where the i-th element stores the reflectance measurement of the captured material at the corresponding wavelength, i.e. $x_{i,t} \in \mathbb{N} = \rho(\lambda_t)$ with $i \in [1, n_1 \cdot n_2]$ and $t \in [1, n_b]$. Following this notation, all the measurements of the i-th element can be extracted along the bands to compose the spectral signature of the captured material, $\mathbf{x}_i \in \mathbb{N}^{n_b} = [x_{i,1}, \ldots, x_{i,n_b}]$. As every material absorbs and reflects the solar radiation at specific wavelengths due its physical-chemical properties, the obtained spectral signature represents the fingerprint of the item, being unique and distinctive for each type of matter. By analyzing this information, we can accurately identify the content of the HSI scene at pixel level. In this thesis, such analysis will be carried out through classification techniques.

In general, the purpose of any classification method is to allocate the target elements into groups/classes based on some kind of criteria or invariant property of those elements. In the case of HSI images, classification methods aim to assign each spectral signature of the data cube to a label, which can be associated or not to a known land cover type [72], by analyzing their spectral, spatial and/or spectral-spatial features, being quite different from the classification of other remote data images. In this context, the classification of HSI data can be considered an optimization problem, where the classifier is defined as a mapping function, which may or may not have some parameters θ, $f(\cdot, \theta) \in \mathscr{F}$ drawn from a hypothesis space $\mathscr{F}$, which describes the relationship between a random feature vector space $\mathscr{X}$ (i.e., the

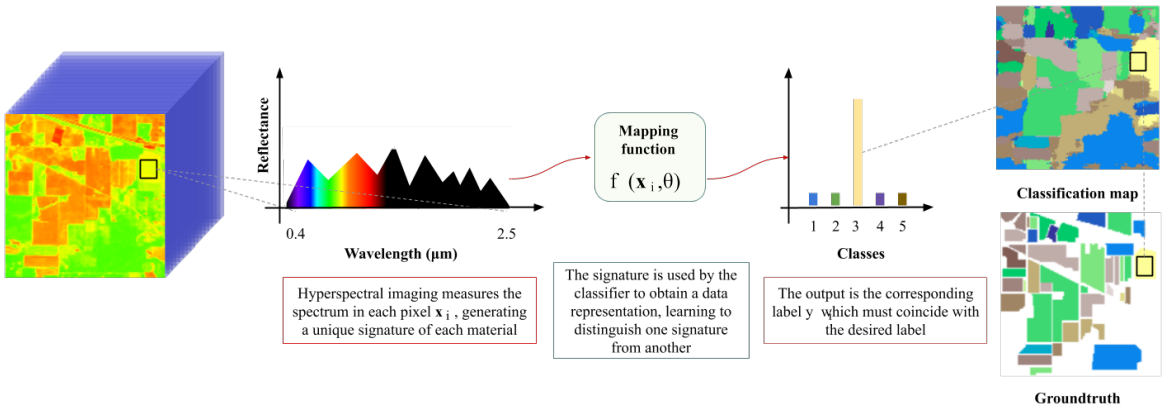


Fig. 3.4 Graphical overview of HSI data classification. The spectral signature of each pixel works like a fingerprint of the captured material. Classifiers exploit that information, analyzing the patterns of each signature to identify the content of the pixel. Asa a result, a classification map is obtained, where each pixel is tagged with the corresponding label of the observed coverage.

instance space with all possible observations) and a random target vector space $\mathscr{Y}$ (i.e., the set of unique and mutually exclusive labels), $f : \mathscr{X} \rightarrow \mathscr{Y}$ [35, 142]. In this case, the instance space $\mathscr{X}$ is the considered HSI dataset $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_{n_1 \cdot n_2}]$ composed by all the spectral pixels, while $\mathscr{Y} \subset \{1, \ldots, n_c\}$ comprises the current n_c labels. Thus, the goal of f is to obtain the sample-label pairs $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{(n_1 \cdot n_2)}$ for each HSI pixel, where $\mathbf{y}_i \in \mathbb{N}^{n_c}$ is the corresponding label of the i-th sample in one-hot encoding. The outcome of applying the mapping function to the entire $\mathbf{X}$ is the classification map $\mathbf{Y} \in \mathbb{N}^{n_1 \times n_2 \times n_b}$, as we can observe in Fig. 3.4.

In the available literature, there is a vast number of scientific contributions about HSI data classification. In general, these methods have been inspired by previous algorithms that were originally developed for statistical pattern recognition and computer vision applications. In this context, these algorithms exhibit a wide variety of methodologies, procedures and different learning strategies. Moreover, they can be classified through various taxonomies depending on the criterion considered. For instance, some interesting works [253, 294, 334] propose a historical line, categorizing the large number of methods and algorithms according to the perspective of the classification problem that has been evolved over the years. Thus, some works emphasize the evolution of the classification problem from a purely statistical point of view, until the development of the automatic learning and the predictive ability of the algorithms [193]. Therefore, the following sections will briefly review the evolution of classification methods, reviewing the fundamental concepts from the point of view of statistical learning (SL) to deep learning (DL) models, through the evolution of pattern recognition (PR) and machine learning (ML).

3.2.1 Classification problem: from statistical learning to deep learning

Within the field of artificial intelligence, the problem of automatic processing and learning has always been a hot topic, developing a large variety of artificial systems in order to analyze and learn the information contained into any type of dataset (such as numerical, text and image databases). In fact, data classification challenge has been addressed from different points of view as algorithms and processing methods have evolved. In this sense, one of the first approaches was given by SL theory [46, 324], which laid many of the fundamentals on which PR and ML fields are currently based. In fact, SL can be observed as the theoretical branch [327].

Following previous notation, SL states that the goal of the mapping function $f(\cdot)$ is to mimic the joint distribution that relates the instances to their respective labels, i.e. $P(\mathbf{X}|\mathbf{Y})$. To do this, a loss function $\mathscr{L}$ (such as the mean squared error –MSE–) is defined to penalize the distance/difference between obtained predictions $f(\mathbf{x})$ and real labels $\mathbf{y}$, for any $(\mathbf{x},\mathbf{y}) \sim P$, computing the *expected risk* $R(f)$ as:

$$R(f) = \int \mathscr{L}\left(f(\mathbf{x}),\mathbf{y}\right) \mathrm{d}P(\mathbf{x},\mathbf{y}) \tag{3.4}$$

If the joint probability P is known, then it is easy to find the hypothesis that best fits the data through Eq. (3.4) as $f^* = \arg\min_{f\in\mathscr{F}} R(f)$. However, P is usually unknown, so Eq. (3.4) becomes impractical. Instead of $R(f)$, the *empirical risk* $\hat{R}(f)$ is employed. Regarding this, if we have access to a training set $\mathscr{D}_{train} = \{\mathbf{x}_i,\mathbf{y}_i\}_{i=1}^n$ which contains some information about P, such as $(\mathbf{x}_i,\mathbf{y}_i) \sim P$, $\forall i \in [1,n]$, we can compute the empirical risk denoted by $\hat{R}(f)$ as:

$$\hat{R}(f) = \int \mathscr{L}\left(f(\mathbf{x}),\mathbf{y}\right) \mathrm{d}P_\delta(\mathbf{x},\mathbf{y}) \tag{3.5a}$$

$$\text{symplified as } \hat{R}(f) = \frac{1}{n}\sum_{i=1}^{n} \mathscr{L}\left(f(\mathbf{x}_i),\mathbf{y}_i\right) \tag{3.5b}$$

where $P_\delta(\mathbf{x},\mathbf{y})$ denotes the *empirical distribution* based on Dirac measure δ:

$$P_\delta(\mathbf{x},\mathbf{y}) = \frac{1}{n}\sum_{i=1}^{n} \delta\left(\mathbf{x}=\mathbf{x}_i,\mathbf{y}=\mathbf{y}_i\right) \tag{3.6}$$

Considering $\hat{R}(f)$, empirical risk minimization can be employed to find the hypothesis f^* that best fits the data as $f^* = \arg\min_{f\in\mathscr{F}} \hat{R}(f)$. As we can observe, this SL approach has one important weakness: it is not ensured that f^* generalizes outside the data [279]. In this context, we can roughly suppose that SL focuses more on the theoretical framework and how

mathematically formalize inference process than on the automatic learning of algorithms, methods, and models.

The second approaches emerged between the 1970s and 1980s, leading to *pattern recognition* (PR), which can be defined as the ability to recognize specific patterns or regularities and respond appropriately to them [286]. Its origins are based on statistics and engineering [48, 90], providing some of the most important milestones in the data analysis [58] such as the nearest neighbor decision rules [77, 113], the Bayes decision theory [69, 223, 313], the implementation of different error rates estimations such as the leave-one-out method [109, 187], the feature evaluation through statistical distances and error bounds [167, 209], the probability density estimation in non-parametric problems [252], the Fisher linear discriminant analysis and multicategory generalizations [90, 210, 356], the decomposition of mixture densities for unsupervised learning [108, 254], the k-means algorithm [163] and its variants [182, 207] and the supervised parameter estimation [85, 160], to name a few. Many applications have exploited the potential of PR methods to automatically extract and process information such as robotics [329], medicine [66, 194], economic forecasting [104], election statistics [206], social media analysis and language processing [15, 68, 221], among others. Also, PR algorithms have been successfully employed to the characterization of those measurements taken from the Earth's surface [58, 107]. However, due to their engineering background, PR does not really imply learning the patterns. They can process the information from the data either by using the knowledge already acquired or by analyzing the statistical information extracted from the patterns and/or their representation, which allows the implementation of traditional handcrafted algorithms that do not even learn from the data.

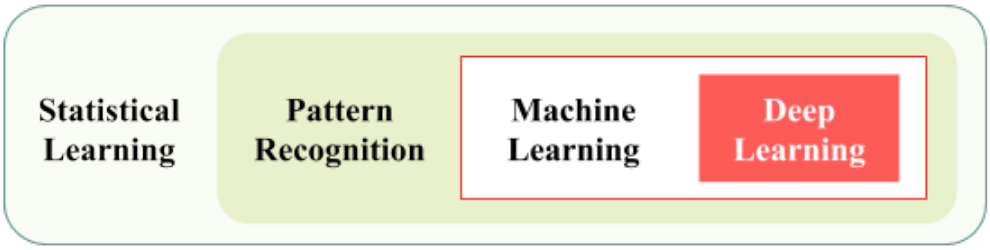


Fig. 3.5 Graphic description of the relationships between statistical learning (SL), pattern recognition (PR), machine learning (ML) and deep learning (DL). Broadly speaking, PR is an engineering application of ML. ML is a form of PR focuses on training machines to recognize patterns and apply them to practical problems. DL can be understood as a subfield of ML, where the models are characterised by their hierarchical and deep structure. Finally, SL gives mathematical support to all the above approaches.

In the 1990s, more emphasis was placed on the learning process by considering statistical and probabilistic techniques, giving rise to the *machine learning* (ML) field, which is more

focused on allowing an artificial system to learn from data rather than through explicit programming. Indeed, the main objective of ML is to design general purpose methodologies and procedures to extract valuable information from the data, such as data regularities, hidden patterns and internal relationships, without it being mandatory to have domain-specific expertise [84]. Although its origins are based on computer science instead engineering, ML is closely connected to PR (see Fig. 3.5), in fact all ML methods are applied in PR systems but not all PR algorithms use ML models [34, 106]. In this regard, ML involves learning patterns from available data in order to apply the knowledge acquired on new data to make predictions about them, sacrificing a little precision in exchange for the automatic learning. There is a broad variety of processing models and methodologies for HSI data classification based on ML approaches [180, 181]. Regarding this, the current literature offers us a fair amount of works that review these methods in detail [1, 8, 14, 46, 62, 72, 99, 114, 117, 202, 204, 224, 262, 302, 360], among others. Very briefly, some of the most widely used algorithms can be grouped into the following categories:

- *Unsupervised* and *supervised* methods: Both are learning strategies in which, in the first case, the methods do not need a guide to learn the internal relationships and hidden patterns of the data, discovering the information contained by themselves, while in the second case, the models need a supervision technique in the form of guided training with labeled data. Some popular unsupervised methods are k-means [135], iterative self-organizing data analysis technique (ISODATA) [332], k-nearest neighbors (KNN) [47] or similarity-based measurements [89], for instance, while random forests (RFs) [130] and support vector machines (SVMs) [226] are very popular supervised methods.

- *Spectral*, *spatial* and *spectral-spatial* methods: These methods differ in the type of features they use to discriminate between one class and another [24]. For example, spectral methods (also called pixel-wise methods) use the spectral information contained in the pixel in isolation, while spectral-wise methods also consider the contextual information that surrounds the pixel. For instance, the spectral angle mapper (SAM) [45, 43] and the traditional distance metrics based classifiers [88, 172] are quite known spectral methods, while morphological profiles (MP) [30, 98, 157] and its variant [82] or the sparse coding (SC) [56, 345] can employ spectral and spatial features.

- *Parametric* and *non-parametric* methods: these categories include those methods that do or do not make some assumptions about the form of the mapping function $f(\cdot)$. In this sense, parametric models assume the data is from a known distribution, defining a fixed structure about $f(\cdot)$ and employing a finite number of parameters. On the contrary, non-parametric models assume the distribution derived from the training

data, without adopting a fixed structure of f, so the number of parameters will depend on the data, being potentially infinite. Maximum likelihood (ML) [185], logistic regression (LR) [12] and linear discriminant analysis (LDA) [88] are some popular parametric models, while evidential reasoning (ER) [290]) and KNN can be considered as non-parametric learning methods.

- *Stochastic* (also *probabilistic*) and *deterministic* classifiers: These categories separate those methods whose output contains a certain degree of uncertainty and those methods whose output is accurately determined by the input. The most widely used probabilistic classifier is the multinomial logistic regression (MLR) [134], while SVMs and extreme learning machines (ELMs) [137] are deterministic, for instance.

However, the ML is a continuously growing field, where new and improved algorithms and methodologies are constantly being developed to refine and achieve more accurate results from increasingly larger and more complex data sets. In this regard, promoted by the great technological advances experienced since the early 2000s, ML has endured a remarkable transformation due to the development of new *deep learning* (DL) models [294]. These models are closely inspired by the biological mechanisms of the human brain, developing a number of algorithms based on artificial neural networks (ANNs) [131, 264, 346]. The main idea behind ANNs lies in the extraction of those relevant features from the available data to obtain abstract representations that allow pattern recognition to be performed without prior knowledge of the data. Indeed, ANNs can be observed as universal function approximators, where given a sequence of patterns $\mathbf{x_1} \dots \mathbf{x_n}$ and the sequence of desired responses $\mathbf{y}_1 \dots \mathbf{y}_n$, the system finds the θ parameters (weights) that best fit the relationship $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{n}$, as we can observe in Fig. 3.6. In this sense, ANNs exhibit some attractive properties which make them a good solution for the classification of HSI data within the RS field:

- ANN automatically adjusts its parameters to a new environment as data evolves, without the need for pre-programmed mechanisms.
- ANN can deal with a large variety of data, whether linear or non-linear, diffuse, probabilistic or noisy, for instance.
- ANN systematically performs many operations in parallel, allowing its computational optimization in many-core architectures, such as multicore processors and graphics processing units (GPUs).

In this regard, DL takes advantage of ANNs to implement very deep and complex architectures, which are based on the stacking of many layers composed by neurons [123, 193]. As a

result, a deep-hierarchical architecture is obtained, in which each layer is able to provide a different interpretation of the data it conveys, extracting and learning increasingly abstract features as it progresses through the network. This has fostered the implementation of very deep models (called deep neural networks or DNNs) with a large a variety of architectures and learning strategies [243, 361], becoming an inspiration for the implementation of new and improved HSI data classifiers, marking a clear trend since 2017 [116, 243, 260, 372].

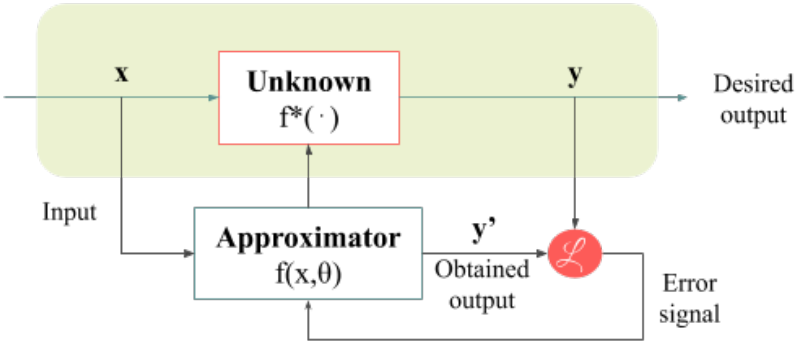


Fig. 3.6 Graphic description of a function approximator. $f^*(\cdot)$ is an unknown and highly complex function that, given an $\mathbf{x}$ input, returns a $\mathbf{y}$ response. $f^*(\cdot)$ can be approximated by a known $f(\cdot,\theta)$ function that, in an iterative way, can adjust its parameters θ by computing an error or loss function $\mathscr{L}(\mathbf{y},\mathbf{y}')$, until the obtained output matches the desired output $\forall \mathbf{x},\mathbf{y}$ in a dataset $\mathscr{D}_{train} = \{\mathbf{x}_i,\mathbf{y}_i\}_{i=1}^{n}$.

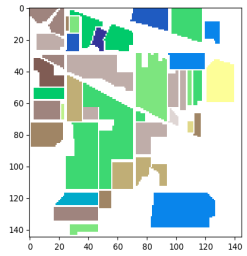
In this sense, the goal of this thesis is to delve into DL techniques applied to HSI data analysis and processing. Along these years of the PhD program, the strengths and weaknesses of several deep classification models have been analyzed, focusing on their performance in RS-HSI data classification tasks, in order to explore and implement new architectures and methodologies that allow the efficient and effective analysis of this type of data, also seeking its optimization on many-core platforms such as GPUs.

3.2.2 Hyperspectral dataset considered in this thesis

At this point it is interesting to note some of the most widely used HSI datasets for classification purposes. These datasets are composed by individual HSI scenes that include labeled samples and unlabeled samples, allowing the implementation of both supervised and unsupervised learning approaches. The labeled samples comprises the scene ground-truth (GT) while the unlabeled samples compose the image background information, being fundamental elements for the training and testing of classification methods. Table 3.3 shows a

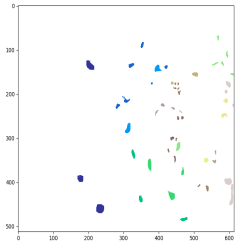
Table 3.3 Details of the ground-truth corresponding to different HSI scenes. In particular, the number of samples per class of the Indian Pines (IP), Kennedy Space Center (KSC), Salinas Valley (SV), University of Pavia (UP) and University of Houston (UH) data sets are provided.

INDIAN PINES (IP)



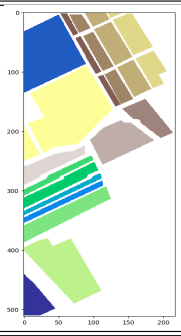
Color	Land-cover type	Samples
	Background	10776
	Alfalfa	46
	Corn-notill	1428
	Corn-min	830
	Corn	237
	Grass/Pasture	483
	Grass/Trees	730
	Grass/pasture-mowed	28
	Hay-windrowed	478
	Oats	20
	Soybeans-notill	972
	Soybeans-min	2455
	Soybean-clean	593
	Wheat	205
	Woods	1265
	Bldg-Grass-Tree-Drives	386
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

KENNEDY SPACE CENTER (KSC)



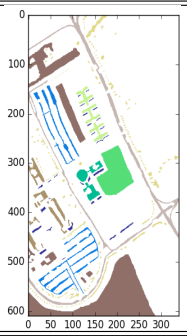
Color	Land-cover type	Samples
	Background	309157
	Scrub	761
	Willow-swamp	243
	CP-hammock	256
	Slash-pine	252
	Oak/Broadleaf	161
	Hardwood	229
	Swap	105
	Graminoid-marsh	431
	Spartina-marsh	520
	Cattail-marsh	404
	Salt-marsh	419
	Mud-flats	503
	Water	927
	Total samples	314368

SALINAS VALLEY (SV)



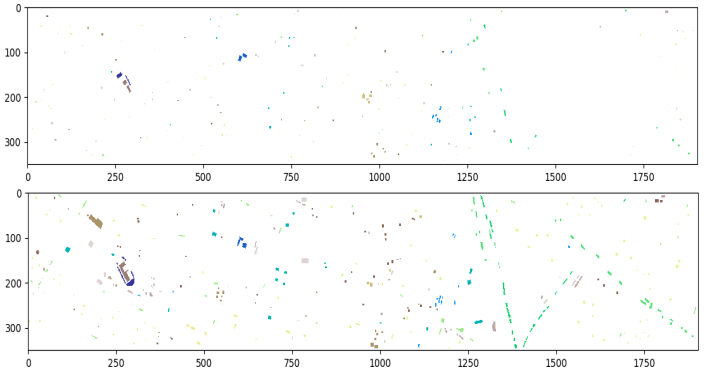
Color	Land-cover type	Samples
	Background	56975
	Brocoli-green-weeds-1	2009
	Brocoli-green-weeds-2	3726
	Fallow	1976
	Fallow-rough-plow	1394
	Fallow-smooth	2678
	Stubble	3959
	Celery	3579
	Grapes-untrained	11271
	Soil-vinyard-develop	6203
	Corn-senesced-green-weeds	3278
	Lettuce-romaine-4wk	1068
	Lettuce-romaine-5wk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

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

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

brief summary of some popular HSI datasets, including the land cover categories with the corresponding number of labeled samples per class.

Some of the most widely used scenes are those scenes captured by the AVIRIS sensor [124], such as *Indian Pines* (IP, also known as 92AV3C), *Kennedy Space Center* (KSC) and *Salinas Valley* (SV) scenes. Focusing on the first one, the IP scene was collected in 1992 over the Indian Pines test site in north-west Indiana. The observed surface (composed by 145×145 pixels) belongs to an agricultural area, being composed by regular patches of different crops coupled with irregular forest and grass zones. Focusing on the spectral features, it collects 224 spectral bands in the wavelength range from 0.4 to 2.5μm with nominal spectral resolution of 10 nm, however, 24 bands were removed in order to avoid null and water absorption bands (in particular [104-108], [150-163] and 220), keeping the remaining 200 bands. This image is particularly challenging because the pixels are very mixed due to a low spatial resolution (it is noteworthy that IP has a high GSD of 20 mpp), and the available GT is composed by 16 classes highly unbalanced. There is also a larger version of IP scene, which is usually known as *big Indian Pines* (BIP). It covers the same agricultural area of IP, but spanning a much larger extent. In particular, BIP comprises 2678×614 pixels with 220 spectral bands. In this case 58 different land cover types constitute the GT, where the 20.33% of the samples are labeled. The second dataset, known as KSC scene, was acquired in 1996 over the Kennedy Space Center, Florida. It contains 512×614 pixels with a spatial resolution of 18 mpp. As IP, the original data cube comprises 224 spectral bands, which were collected with a bandwidth of 10 nm in the spectral range 0.4-2.5μm. However only 176 spectral bands have been retained after the removal of those spectral bands affected by water absorption and low SNR. This dataset is characterized by a very small set of labeled samples, in fact, only 5211 out of the available 314368 pixels are labeled into 13 different land cover classes, i.e. only a 1.66% of the image is labeled in comparison with the 48.74% of IP. The third HSI dataset, known as SV scene, was also collected over several agricultural fields of Salinas Valley, California during a flight campaign in 1998, covering an area of 512×217 pixels with a higher spatial resolution of 3.7 mpp. As IP and BIG, SV records spectral information in the wavelength range from 0.4 to 2.5μm with bandwidth of 10 nm, collecting 224 bands, of which 20 water absorption bands have been removed, in particular [108-112], [154-167] and 224. As in IP, its ground truth is also divided into 16 different land cover categories of different crops and fallows, where the 48.72% of the data is labeled. This scene is usually cropped, selecting a particular sub-scene (Salinas A) composed by 83×86 pixels that represent a difficult classification scenario. In fact, Salinas A contains highly similar pixels, which correspond to the same type of crop with different maturation times [263].

In addition to the scenes provided by AVIRIS, there are other interesting HSI datasets due to the nature of their content. For instance, the ROSIS sensor [186] provided the *University of Pavia* (UP) and *Pavia Centre* (PC) scenes during a flight campaign over Pavia, northern Italy. In particular, UP scene was acquired over the campus of Pavia city and contains 610×340 pixels, while PC scene was collected over the historical city. The original PC image contains 1096×1096 pixels, however, a 381-pixel-wide strip with no information is usually removed, obtaining as a result a 1096×715 scene. Both scenes were gathered with a high spatial resolution of 1.3 mpp. Moreover, ROSIS instrument is able to cover up to 120 spectral bands in the wavelength range from 0.43 to 0.86μm, nevertheless 12 and 13 noisy bands of UP and PC have been removed, keeping the remaining 103 and 102 spectral bands. Both scenes contain 9 different land cover categories, which belong to multiple man-made structures, natural objects and shadows, being 20.62% and 18.90% of UP and PC scenes labeled, respectively.

Also CASI instrument [16] has provided a large HSI dataset, known as University of Houston (UH) [341]. This dataset was presented at the IEEE Geoscience and Remote Sensing Society (GRSS) Image Analysis and Data Fusion Technical Committee during the 2013 Data Fusion Contest (DFC) [83], being one of the most recently acquired HSI data for classification tasks. In fact, UH scene was gathered in 2012 over the university campus and the adjacent residential area located in Houston. It contains 349×1905 pixels with a spatial resolution of 2.5 mpp (it is the second scene with the best spatial resolution, behind of UP and PC scenes, and ahead of SV dataset). It includes 144 channels with the sampled spectral information in the 0.38-1.05μm range. The available GT is composed by 15 different land cover categories, where the amount of labeled data is quite limited. In fact, only the 2.26% of the samples is labeled. Moreover the GT is divided into two spatial disjoint regions, the training region (which is composed by the 0.43% of the labeled data) and test region (which is composed by the 1.83% of the labeled samples), making it an ideal benchmark to test the robustness and generalization of classification methods.

3.3 Deep Learning: overview about basic concepts

3.3.1 Towards deep architectures: the neuron as a starting point

DL proposes a new learning paradigm based on deep and complex neural architectures for the extraction of features at different levels and the learning of abstract data representations. As standard and shallow ANNs (i.e. the multilayer perceptron –MLP– [173]), the basis of DL methods is inspired by the behaviour of biological neuronal models, where a set of neurons

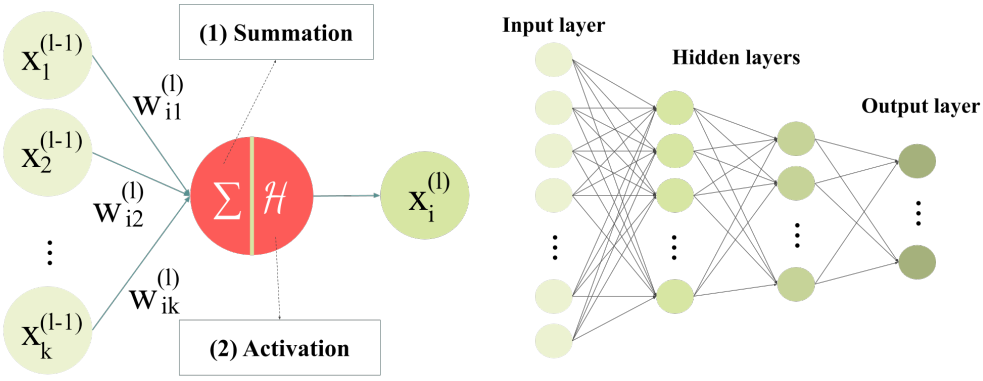


Fig. 3.7 Graphic description of the i-th neuron in the l-th layer. This neuron processes the input features of the previous layer $\mathbf{x}^{(l-1)} = [x_1^{(l-1)}, \ldots, x_k^{(l-1)}]$ by a weighted sum of them, applying its respective weights $\mathbf{w}_i^{(l)} = [w_{i,1}^{(l)}, \ldots, w_{i,k}^{(l)}]$. The result is processed by a threshold-based function $\mathscr{H}$ (the activation function) that decides whether or not the input stimulus activates the neuron to produce the final output. These neurons are organized in layers, producing a large variety of architectures depending on the way they are connected.

can communicate with each other through synaptic connections in order to provide a certain response to a given stimulus. In this context, the main computational element of any artificial neural system is the computation cell or artificial neuron.

The basic functionality of artificial neurons is described in Fig. 3.7[4]. They are interconnected creating a layer-based architecture, where the i-th neuron of the l-th layer receives as input data either the entire vector or a part of the feature vector processed by the previous layer, i.e. $\mathbf{x}^{(l-1)} = [x_1^{(l-1)}, \ldots, x_k^{(l-1)}]$. Each feature is combined with the others through a weighted summation, employing a mechanism of weights on the neuron's connections to simulate the synaptic procedure of the biological system. In fact, these weights represent the significance of each feature in the neuron. The resulting output is processed by a threshold-based function, which decides whether or not to trigger the neuron to produce the final activation output $x_i^{(l)}$. Mathematically, Eq. (3.7) details the processing performed by the artificial neuron:

$$x_i^{(l)} = \mathscr{H}\left(\sum_{j=1}^{k} w_{i,j}^{(l)} x_j^{(l-1)} + b_i^{(l)}\right) \tag{3.7}$$

[4]This thesis focuses on the standard neuron that takes the weighted sum of its input values, and returns the output when it has processed by an activation function, building multilayer feedforward networks (MFNs), such as the MLP, which is trained by forward and backward propagation. In this sense other traditional ANNs, such as the radial basis function (RBF) [31] and Kohonen self-organizing map (SOM) [178] networks are beyond the scope of this work, as well as their training processes.

where $x_j^{(l-1)} \in \mathbf{x}^{(l-1)}$ is the j-th feature of the input data $\forall j \in [1,k]$, $w_{i,j}^{(l)} \in \mathbf{w}_i^{(l)}$ is the weight that connects the i-th neuron in the current layer l with the j-neuron in the previous layer $l-1$ and $b_i^{(l)}$ is a threshold or bias. $\mathscr{H}$ represents the activation function. Finally, $x_i^{(l)}$ is the obtained output of the i-th neuron in the current layer l.

The architecture based on layers of neurons has proven to be more flexible and versatile than standard ML algorithms, such as the SVM, emulating the global and complex mapping function $f(\cdot,\theta)$ through several independent (i.e. the design of each layer is completely isolated from the rest, comprising an arbitrary number of neurons through a bias node) and hierarchically ordered (i.e., the outputs of one $f^{(l)}$ are the input of the next $f^{(l+1)}$) functions such as:

$$f(\mathbf{x},\theta) \approx \hat{f}\left(f^{(L)}\left(\ldots\left(f^{(1)}\left(\mathbf{x},\theta^{(1)}\right)\ldots\right),\theta^{(L)}\right)\right) \tag{3.8}$$

where each $f^{(l)}(\mathbf{x}^{(i-1)},\theta^{(l)}) = \mathbf{x}^{(l)}$ represents the mathematical transformations that the l-th layer applies to its input feature data $\mathbf{x}^{(i-1)}$ taking into account its parameters $\theta^{(l)}$, $\forall l \in [1,L]$ (being L the number of layers in a neural-based model), and $\hat{f}(\mathbf{x}^{(L)}) = \mathbf{y}$ is the final classification layer, which takes the obtained abstract data representation and assigns its corresponding label [243]. More precisely, and taking into account the Eq. (3.7) and that the original input data is composed by the set of HSI instances $\mathbf{X}$, each layer can be defined by the following Eq. (3.9):

$$\mathbf{X}^{(l)} = \mathscr{H}\left(\mathbf{X}^{(l-1)},\mathbf{W}^{(l)},b^{(l)}\right), \tag{3.9}$$

where $\mathbf{X}^{(l-1)}$ and $\mathbf{X}^{(l)}$ represent the neuronal activations of the previous and current layer, respectively, $\mathscr{H}$ is the activation function, $\theta^{(l)} = \mathbf{W}^{(l)},b^{(l)}$ are the layer's parameters, i.e. the matrix of weights $\mathbf{W}^{(l)} = [\mathbf{w}_1^{(l)},\ldots,\mathbf{w}_K^{(l)}]$ where each $\mathbf{w}_i^{(l)}$ defines the strength of connection between the i-th neuron in layer l and all or some of the neurons in the previous layer, $\forall i \in [1,K]$, being K the number of neurons comprised by layer l. In this way, the neural model can capture really complex polynomial relationships as the data is processed by the stack of layers, which would be almost impractical to convey by other ML methods.

3.3.2 Where learning resides: backward step and parameter updating

Neural architectures can be divided into *feedforward* and *recurrent* models according to the way neurons in different layers are connected. Feedforward models always connect neurons in layer l to those neurons in layer $l+1$, with no feedback connections in which the outputs of a layer l feed back to the layer itself or to previous layers, i.e. the data flow always goes ahead from one layer to another. In contrast, recurrent networks include feedback connections between their layers. In any case, the data always flows through the network

until it reaches the last classification layer, applying Eq. (3.8). This step is known as the forward step or *forward propagation*, where the model's parameters (i.e. weights $\mathbf{W}^{(l)}$ and biases $b^{(l)}$, $\forall l \in [1, L]$) are applied over the data to extract the relevant and discriminative information. In this sense, the appropriate adjustment of these parameters is fundamental for the performance and reliability of the model. It is precisely the parameter adjustment procedure where the network learns to extract the most relevant information from the data that will best contribute to its performance during classification.

As we can observe in Fig. 3.6, the learning process of any NN takes the inputs $\mathbf{x}$ and the desired outputs $\mathbf{y}$ and updates the internal state (i.e. the model's parameters) according to a loss function $\mathscr{L}$ that measures the distance between the predicted output and the desired one, with the aim of reducing the difference between them. Eq. (3.10) provides a popular non-negative loss function, known as the MSE:

$$\mathscr{L}(\mathbf{W}, \mathbf{b}) = \frac{1}{2n} \sum_{i=1}^{n} ||\mathbf{y}'_i - \mathbf{y}_i||^2 \tag{3.10}$$

where $\mathbf{W}$ and $\mathbf{b}$ denote the collection of all weights and biases of the model, respectively. In order to obtain the set of $\mathbf{W}$ and $\mathbf{b}$ that minimize the loss function, the neural model usually applies some optimizer based on the gradient of the loss function, which can measure the impact of modifying each model parameter on the error achieved (*error-change rate*), which is given by the partial derivative of the loss function with respect to any weight or bias of the network, i.e. $\partial\mathscr{L}/\partial w_{i,j}^{(l)}$ and $\partial\mathscr{L}/\partial b_i^{(l)}$. However, the estimation of the error-change rate for each parameter cannot be directly obtained, as the layers are connected in a hierarchical way, so any change in a parameter can affect the following layers, and therefore their corresponding parameters. In this context, the entire optimization process is conducted through an iterative process using the *backpropagation* algorithm [284], which decomposes the derivative calculation and propagates back the error signal from the end of the model to the initial layers.

Considering $z_i^{(l)}$ as the intermediate result of the i-th neuron located at the l-th layer (i.e. the result of the sum of the Eq. (3.7) without the activation function), the procedure defines the *local error gradient* of the i-th neural unit at the l-th layer as:

$$\delta_i^{(l)} = \frac{\partial\mathscr{L}}{\partial z_i^{(l)}} \tag{3.11}$$

In this sense, the backpropagation methods begins to propagate the error signal through the neurons in the last layer of the network through the chain rule. As we can observe in Fig. 3.8, the gradient of the lost function can be directly obtained at the neurons outputs as $\partial\mathscr{L}/\partial x_i^{(L)}$,

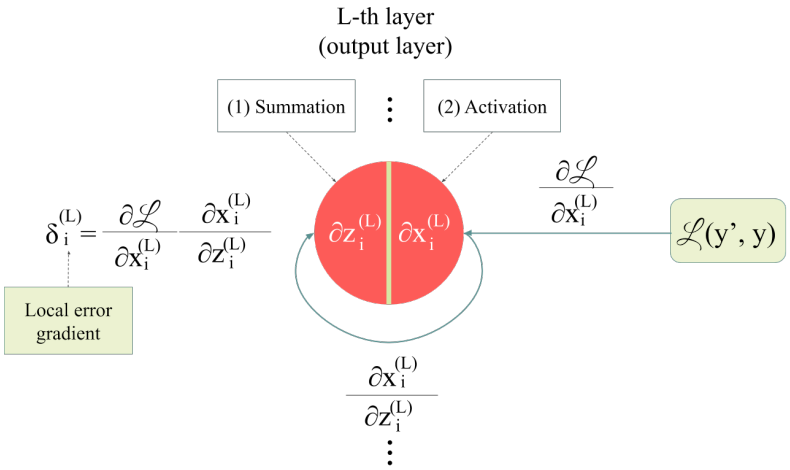


Fig. 3.8 Local error gradient at the i-th neuron in the L-th layer (output layer). The error signal given by the loss function $\mathscr{H}$ is backpropagated to each neuron of the output layer.

where $x_i^{(L)}$ is the post-activation output. Then, in order to backpropagate the gradient signal to the pre-activation state $z_i^{(l)}$, the intermediate derivative $\partial x_i^{(L)} / \partial z_i^{(L)} = \mathscr{H}'(z_i^{(L)})$ is obtained. Finally, the desired local error gradient is obtained as:

$$\delta_i^{(L)} = \frac{\partial \mathscr{L}}{\partial x_i^{(L)}} \mathscr{H}' \left(z_i^{(L)} \right) \tag{3.12}$$

Once the local error gradient is obtained at each neuron of the last layer, the signal can be easily backpropagated through the layers as we can observe in Fig. 3.9, where the local error gradient of the current neuron can be calculated as the sum of the next layer's gradient signals with the first derivative of the activation function pondered by the weights of the connections. Eq. (3.13) gives us the mathematical formulation:

$$\delta_i^{(l)} = \sum_j w_{j,i}^{(l+1)} \delta_j^{(l+1)} \mathscr{H}' \left(z_i^{(l)} \right) \tag{3.13}$$

Finally, as the error signal is propagated throughout the network, from the top layers (i.e. those layers that are more close to the output data) to the bottom layers (i.e. those layers that are more close to the input data), the error change rate of each weight $w_{i,j}^{(l)}$ and bias $b_i^{(l)}$ can be obtained applying the chain rule as:

$$\frac{\partial \mathscr{L}}{\partial w_{i,j}^{(l)}} = \delta_i^{(l)} x_j^{(l-1)} \quad \text{and} \quad \frac{\partial \mathscr{L}}{\partial b_i^{(l)}} = \delta_i^{(l)} \tag{3.14}$$

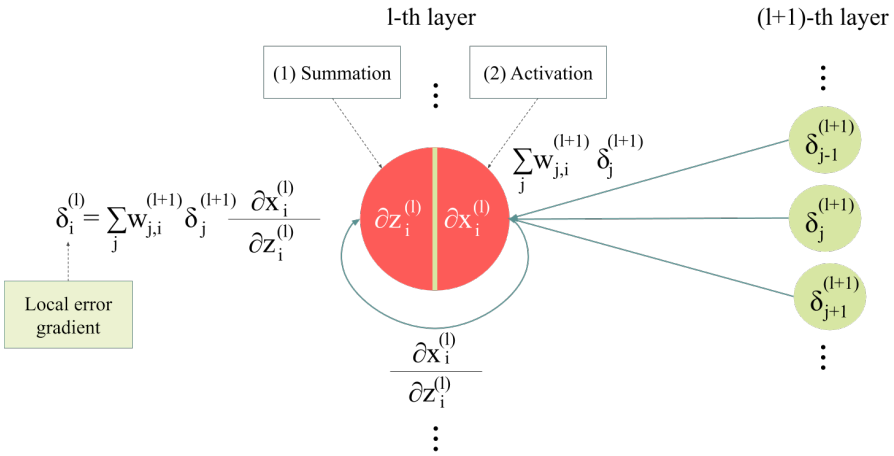


Fig. 3.9 Local error gradient at the i-th neuron in the l-th hidden layer.

The learning of the network is completed with the parameters adjustment following the gradient descent of the error function provided by Eq. (3.14) in order to reduce the distance between the predicted and desired outputs. In this sense, each weight $w_{i,j}^{(l)}$ and bias $b_i^{(l)}$ is updated by Eq. (3.15):

$$w_{i,j}^{(l)} = w_{i,j}^{(l)} - \varepsilon \frac{\partial \mathscr{L}}{\partial w_{i,j}^{(l)}} \quad \text{and} \quad b_i^{(l)} = b_i^{(l)} - \varepsilon \frac{\partial \mathscr{L}}{\partial b_i^{(l)}} \tag{3.15}$$

where the hyperparameter ε indicates the learning rate, whose value must be chosen so it does not exceed the loss function (very high learning rates) but neither to extremely slow down the convergence of the optimizer (very low learning rates).

As we can observe, the entire forward-backward propagation scheme allows networks to learn directly from data, resulting in a more compact and efficient learning procedure, which is completely oriented to the task conducted by the network.

3.3.3 The relevance of feature extraction in deep architectures

If the backward step is essential for the network's learning, the forward step is not less important. In fact, the stack of hierarchical functions defined by Eq. (3.8) of any ANN and/or DNN model allows to exploit the ability of neural layers to extract data representations at different levels during the step forward, where the lower layers extract more basic and generic characteristics, such as edges, corners and textures, and the deeper layers gradually refine those basic characteristics to obtain more complex and high-level structures, building more abstract and discriminating data representations. In this sense, the functions $f^{(1)} \dots f^{(L)}$ work

together to become a feature extractor (FE) that encodes the most informative features from the input data, while the final $\hat{f}$ assigns the label to the original input data by analyzing its obtained features (i.e. the high-level data representation).

FE and ML/DL have always gone hand in hand. Traditionally, ML classification methods have been enhanced by handcrafted FE approaches, with the aim of providing representative features for the final classifier [272], as we can observe in Fig. 3.10. Usually, these features are manually designed by the user through some popular spectral-spatial FE methods such as the speeded-up robust features (SURF) [22, 161], the histogram of oriented gradients (HOG) [81], the principal component analysis (PCA) [282, 337], the gradient local auto-correlation (GLAC) [57, 176], the local binary patterns (LBP) [203], the scale invariant feature transform (SIFT) [7] or the newest invariant attribute profiles (IAPs) [153], among others. However, these handcrafted features are very specific and do not usually provide a sufficiently abstract representation of the data. Moreover, they are critically dependent on the developer/user's knowledge, making hard to obtain their most optimal configuration [344].

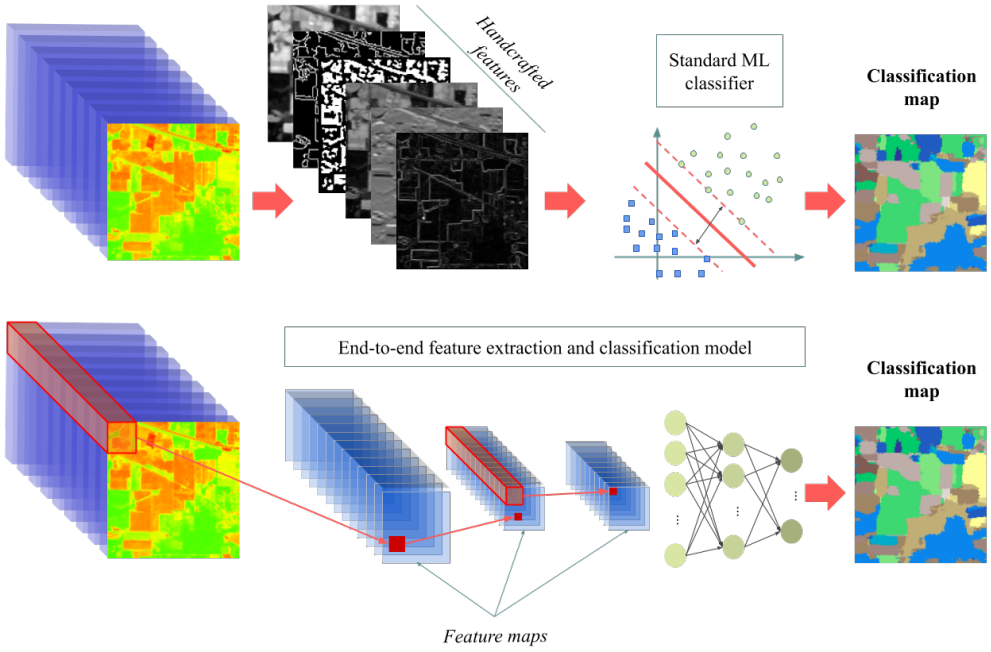


Fig. 3.10 Comparative between hadcrafted features exploited by standard ML algorithms and features automatically extracted and classified by deep neural models.

On the contrary, neural-based models are able to automatically extract features from the raw input data, shifting the burden of feature design by developing an end-to-end framework composed by both FE-stage and final classification stage, as Fig. 3.10 represents. Moreover, as the networks are trained with the same data considered in the classification problem, the FE-stage is able to learn highly abstract features that are directly obtained and refined by

the classification problem itself, allowing a higher adaptability and flexibility in comparison with standard handcrafted features [367]. As a result, neural-based models have attracted a significant attention from the ML scientific community, thanks to their great potential for capturing patterns and structural relationships within the data, demonstrating a better recognition performance than some traditional FE methods [9, 29, 219, 265]. In addition, technological advances (both in hardware devices and software frameworks) within the field of computing and telecommunications have allowed the development and implementation of much deeper architectures and more complex layers than those implemented by traditional ANNs, fostering the rise of the DL architectures [241, 243, 334, 361].

In this regard, it can be said that *depth* is the main characteristic of deep models, since the greater the depth, the greater the level of abstraction. Then, the following question may arise: at what point a model goes from being shallow to deep? Similar to the definition of RS, there is no general accepted agreement on the depth a network must have to be considered as a deep model [294]. However, some experts in the DL field consider single-hidden-layer models as shallows ANNs, being considered DNN when they have two or more hidden layers [26, 150, 183]. Some authors even distinguish between deep, very deep (VDNNs) [304, 310, 311] and ultra-deep (also known as extremely deep neural networks, EDNNs) [191] models, depending on whether they have more than twelve or fifty hidden layers respectively, reaching even thousands of layers [141]. Table 3.4 provides a brief summary of these categories.

Table 3.4 General taxonomy of deep neural-based models depending on their depth

	Shallow model	**Deep model**	**Very deep model**	**Ultra-deep model**
Hidden layers	1	2, 3+	12+	50+

Although it is an important element, depth is not everything within the DL field. In fact, the most powerful characteristic of deep methods is the *handling of the features* extracted from the original data, which allows for complex data representations during the FE stage than traditional shallow models [321]. This has a major impact when dealing with HSI data. Fig. 3.11 provides a graphic description of those features that can be exploited in a HSI data cube. As we can observe, the HSI data cube $\mathbf{X} \in \mathbb{N}^{n_1 \times n_2 \times n_b}$ not only offers the spectral information contained in each of its pixels thanks to its spectral domain given by n_b, but also the contextual information of them, thanks to its spatial organization as a map of $n_1 \times n_2$ pixels. This allows to extract spatial relationships between a pixel and its neighbors [188]. These types of features have already been widely exploited by traditional shallow ML models (for instance, through any of the FE method mentioned above) [98, 317]. However the novelty of the DL lies in the fact that DNNs can easily adapt their architectures to efficiently

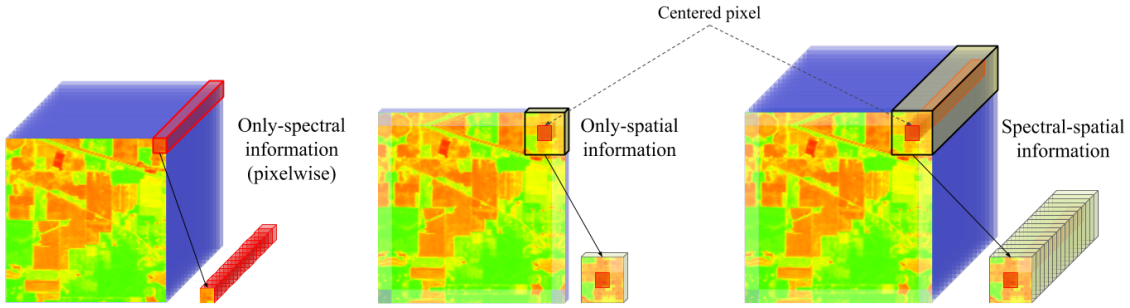


Fig. 3.11 Graphical representation of the information that can be processed by an HSI classifier: spectral, spatial, and spectral-spatial features are extracted from the original HSI data cube and processed by the classification method. For spatial methods it is very common to use a mirroring strategy over the edges of the image to take advantage of the pixels located at the bordes, otherwise these pixels are cut out and not taken into account during the classification process.

exploit and even combine both types of information in a more versatile end-to-end fashion. For instance, shallow ANNs usually employ spectral features by processing each pixel of the HSI image in isolation due to their fully-connected architectures.

In this sense, the topological structure of DNNs is closely linked to both the type of features that can be extracted and the way they are processed. For instance, pixel-wise DNN models are usually composed by *fully-connected* (FC) layers as traditional ANNs. This type of layer allows the extraction of spectral features, reshaping the HSI data cube as a matrix of $(n_1 n_2) \times n_b$ dimensions. Each spectral pixel is individually processed by the layers [285], learning the spectral relationships between the information contained in the pixel (i.e. the spectral signature) and the land cover type it represents [62]. The effectiveness of these methods depends on the spectral quality of the pixel, achieving very accurate results when each pixel contains a perfect signature of a single cover material [103]. However, the technical limitations of the sensor do not allow the collection of data with a high spatial resolution. So in real scenarios, the pixels are very mixed and can contain:

- Small sub-pixel objects.
- Boundaries between different land cover materials.
- Marked transition zones between different ecological components (ecotone).

As a result, the HSI data cube suffers from high intra-class (also known as intra-cluster) variability and great inter-class (or inter-cluster) similarity, in the sense that pixels belonging to the same class may contain very different spectral signatures due to the mixture of materials and elements, while pixels belonging to different classes may present very similar spectral

signatures if they are in border areas, for instance. In this regard, the performance of pixel-wise DNN strongly depends on the available samples, consuming a large amount of them to properly cover these variations.

To overcome these limitations, some works take into account the spatial information during the classification task [157, 164, 362]. The main assumption is that neighboring pixels usually belong to the same land cover category [115, 233], so they can provide valuable additional information that can reduce the high variability within the data, managing also the label uncertainty. Thus, spatial-based classifiers consider the contextual information provided by the pixel neighborhood window [32, 61], which can be extracted by traditional FE methods, such as principal components [166, 282, 337], covariance matrices [143], minimum noise fraction (MNF) [357], Gabor filtering [64, 169], among others. In particular, the use of PCA is the most widespread method due to its simplicity, reducing the spectral dimension of the data (which also reduces the spectral redundancy and band correlations) while keeping the spatial information [132, 247]. It is noteworthy that methods such as PCA do not eliminate spectral information, but rather compress it, so some authors do not consider these models to be purely spatial while others do [61, 136, 218]. On the other side, this spatial information can be processed by the DNNs by FC architectures (which vectorize the information) or by *convolutional* architectures [132, 247] (which allow to maintain the original shape by applying 2D and 3D convolutions layers).

Spatial-based methods does not generally guarantee the desired improvement in classification performance compared to spectral methods. In this context, the exploitation of spatial and spectral features together is more beneficial than in isolation, as it comprises both the analysis of spectral signatures and the associated spatial-structural information [242, 246, 352]. In this sense, DNN models follow two main strategies in order to process both kind of features: i) the first one is inspired by the traditional ML vector vision [62, 63], where the spatial information is vectored and included as concatenated information to the spectral vector, and ii) the second strategy is inspired by the processing of the 3D patches extracted from the original HSI data cube in order to maintain the original structure information [61, 242, 245, 246], processing the spectral-spatial information in a more natural and intuitive way. In the last case, convolutional neural networks (CNN) [193] are employed instead the standard FC architectures.

3.3.4 New layers for better feature extraction

Supported by the development of image processing techniques [193] and as a direct consequence of the different types of features to be processed (spectral, spatial and spectral-spatial), deep models have adapted their architectures looking for the optimal FE stage. As a result, a

large variety of layers have been developed. In this regard, Paoletti et al. [243] performs a detailed review about the most popular ones.

Fully connected layer

It is the most popular layer of traditional ANNs, being widely used since the 1960s in models such as the MLP [74, 110]. As we can observe in Fig. 3.12, each neuron located in the l-th layer is connected to the outputs of the neurons in the previous layer $l-1$ through weighted connections. In this sense, the FC layer only applies a linear dot product between the input layer data, which is arranged into a vector $\mathbf{x}^{(l-1)}$, and the matrix of weights that connect each neuron to the input $\mathbf{W}^{(l)}$. An offset or bias is usually added to this product so that the neuron does not overflow the activation function afterwards, $b^{(l)}$.

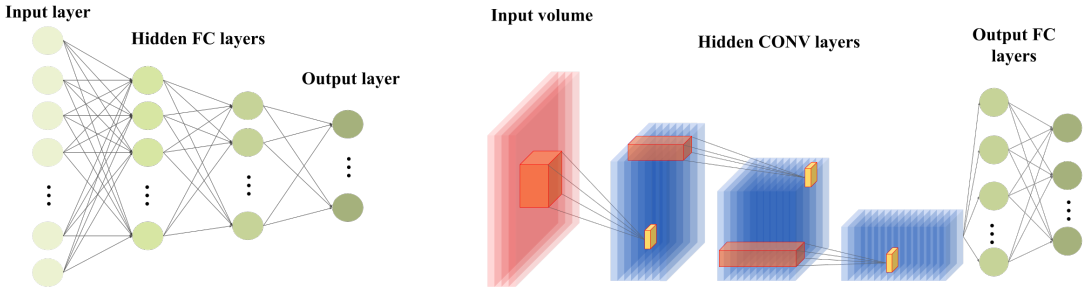


Fig. 3.12 Graphical comparison between fully connected (FC) and convolutional (CONV) models. The FC layers are characterized as an array of neurons where each neuron of the l-layer is connected to the neurons of the $l-1$ layer. They process vectorized information, making them ideal for processing the spectral information contained in the HSI cube. On the contrary, the CONV layers are organized in kernels locally connected to certain regions of the entrance, being organized in 1D, 2D and 3D structures. As a result they can process information arranged in vector, matrix and volume structures, respectively, being ideal to process the spectral-spatial information contained in the HSI cube.

When working with FC-based achitectures, it is very common to reshape the HSI data cube into a matrix $\mathbf{X} \in \mathbb{N}^{(n_1 n_2) \times n_b}$. Thus, when it is sent to the network, each layer performs a matrix multiplication between the input matrix $\mathbf{X}^{(l-1)}$ and its own weights $\mathbf{W}^{(l)}$, so each mapping function $f^{(l)}$ given above by Eq. (3.9) is defined as:

$$\mathbf{X}^{(l)} = \mathbf{W}^{(l)} \cdot \mathbf{X}^{(l-1)} + b^{(l)} \tag{3.16}$$

The FC-based architecture imposes a large number of parameters to train due to the large number of connections between each layer, so they are very prone to over-fitting problems when there is not enough training data. Also, the reshaping of input data into vectors greatly

restricts the representative capacity of the model, losing the spatial relationships of the data and its potential to reduce the model's uncertainty during classification [60]. In this regard, the FC-based architecture can be ineffective for spatial and spectral-spatial feature analysis.

Convolution layer

The first works about convolution (CONV) layers appear in the late 1980s [192] as a neural model inspired by the ventral pathway of the visual cortex, whose neurons are activated or not depending on some specific visual stimuli that fall within their range (also known as local receptive fields). These neuronal activations produce a series of responses in a hierarchical way, so deeper neurons of the visual cortex will be able to respond to more abstract stimuli.

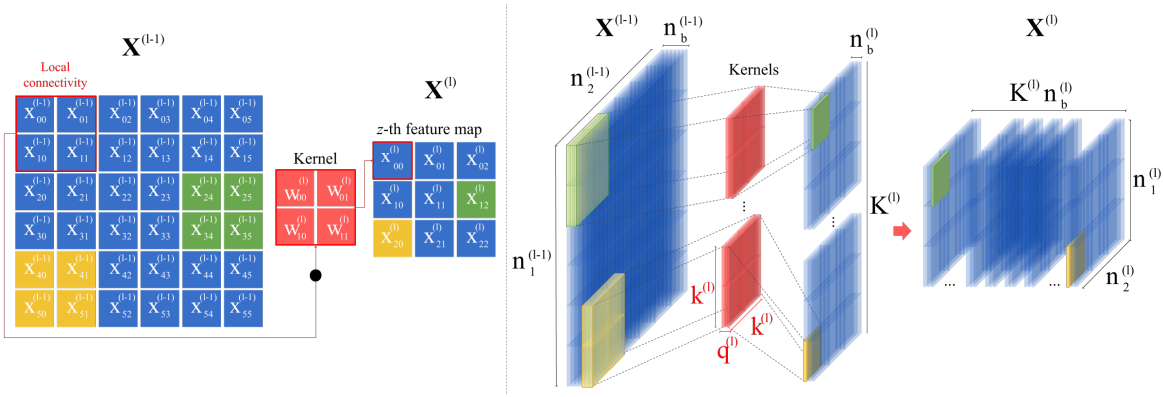


Fig. 3.13 Graphical representation of a convolution layer. On the left side, we can observe how a 2×2 spatial kernel is applied over a 6×6 input slide. The kernel is sliced with a stride of 2, moving every two columns/rows. As a result, a 3×3 feature map is obtained. This behavior can be extended to multidimensional input data (on the right side), so the kernel slides through the spatial and spectral dimensions with a local receptive field defined by $k^{(l)} \times k^{(l)} \times q^{(l)}$. The resulting feature maps are stacked together, resulting in an output feature volume.

In this regard, the CONV layer defines a *filter bank* similarly to the local receptive field, being able to process multidimensional arrays by applying a linear n-dimensional kernel over small pre-defined regions from the layer's input data. As Fig. 3.13 illustrates, the l-th CONV layer performs a FE stage over the input data (denoted as $\mathbf{X}^{(l-1)}$) by mimicking the operation of a sliding window algorithm. Particularly, it defines $K^{(l)}$ fixed-size filters that can be both spectral and spatially overlapped over the input $\mathbf{X}^{(l-1)}$, performing a dot product between the kernel's weights and the current features window. These filters slide through both dimensions by a stride $s^{(l)}$, obtaining in each operation a value of the output features map. Unlike the FC layers, the structure of the filters can be 1D, 2D and 3D, depending on the layer's filter bank, so it can also accept input data organized in 1D, 2D, and 3D arrays, allowing for the

extraction of spectral, spatial and spectral-spatial features in a natural and structured way. For instance, let consider an input volume $\mathbf{X}^{(l-1)} \in \mathbb{R}^{n_1^{(l-1)} \times n_2^{(l-1)} \times n_b^{(l-1)}}$, the filter bank can be defined as a $k^{(l)} \times k^{(l)} \times q^{(l)}$ kernel, where $k^{(l)}$ refers to the spatial dimensions and $q^{(l)}$ to the spectral one. The resulting output $\mathbf{X}^{(l)} \in \mathbb{R}^{n_1^{(l)} \times n_2^{(l)} \times K^{(l)} n_b^{(l)}}$ will be denoted as feature volume (also known as feature vector or feature map depending on its dimensions). In this sense, each mapping function $f^{(l)}$ previously defined by Eq. (3.9) is redefined as indicated by Eq. (3.17a), while Eq. (3.17b) illustrates the computation of the output element (i, j, t) of the z-th filter (being $z = \{1, \cdots, K^{(l)}\}$) that belongs to the l-th convolution layer:

$$\mathbf{X}^{(l)} = \left(\mathbf{W}^{(l)} * \mathbf{X}^{(l-1)} + b^{(l)} \right)_{K^{(l)} \times k^{(l)} \times k^{(l)} \times q^{(l)}} \tag{3.17a}$$

$$x_{i,j,zt}^{(l)} = \sum_{\hat{i}=1}^{k^{(l)}} \sum_{\hat{j}=1}^{k^{(l)}} \sum_{\hat{t}=1}^{q^{(l)}} \left(w_{\hat{i},\hat{j},\hat{t}}^{(l)} \cdot x_{(i \cdot s^{(l)} + \hat{i}),(j \cdot s^{(l)} + \hat{j}),(t \cdot s^{(l)} + \hat{t})}^{(l-1)} \right) + b^{(l)} \tag{3.17b}$$

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 $i \in [1, n_1^{(l)}]$ and $j \in [1, n_2^{(l)}]$. In this sense, $n_1^{(l)}$ and $n_2^{(l)}$ are the spatial dimensions of the resulting output volume $\mathbf{X}^{(l)}$, which can be obtained from the input volume's dimensions, $n_1^{(l-1)}$ and $n_2^{(l-1)}$, as:

$$n_1^{(l)} = \left\lfloor \frac{n_1^{(l-1)} + 2p^{(l)} - k^{(l)}}{s^{(l)}} \right\rfloor + 1 \quad \text{and} \quad n_2^{(l)} = \left\lfloor \frac{n_2^{(l-1)} + 2p^{(l)} - k^{(l)}}{s^{(l)}} \right\rfloor + 1 \tag{3.18}$$

being $s^{(l)}$ the stride and $p^{(l)}$ the zero-padding that fills the edges of the input volume with zeros. t and $\hat{t}$ are the spectral indices that cover the data and weight volumes along the channel dimension, obtaining t in a quite similar way to i and j, i.e. $t \in \left[1, n_b^{(l)}\right]$, where $n_b^{(l)} = \left\lfloor n_b^{(l-1)} - q^{(l)}/s^{(l)} \right\rfloor + 1$, spectral zero-padding is not usually employed, while the spectral stride is usually set to 1. Index $z \in [1, K^{(l)}]$ indicates the filter that is currently applied over the input data.

During the training step, the CONV layer adjusts its weights to learn some features distributed in a certain way both in spatial and spectral domains (depending on the dimension of the kernel), applying them as a filter to other regions of the input data [101]. Focusing on HSI data processing, the local connectivity allows the CONV layer to learn spectral-spatial correlations and regularities among HSI pixels, overcoming the limitations of FC-based architectures. Moreover, the sparse connectivity coupled with the parameter sharing mechanism significantly reduce the number of parameters that must be adjusted within a CONV layer compared to a FC layer.

Pooling layer

Pooling (POOL) layers arise from the popularization of CONV-based architectures (i.e. CNN models) for image processing. In this context, these layers are more focused on the spatial processing of the data than on the spectral or spectral-spatial one. Indeed, they arise as an effort to overcome the great sensitivity of the CONV layer to the spatial location of the features, since they reduce the spatial resolution of the output volume $\mathbf{X}^{(l)}$ by performing a non-linear sub-sampling strategy. Initially, this limitation was addressed through a plain down-sampling conducted by strided CONV layers [308]. The spatial down-sampling of feature maps not only allows for a more robust data representations, reducing the impact of small spatial shifts (invariance to spatial translations), but also summarize the spatial features into a smaller volume, reducing the model's complexity in terms of volume's dimensionality and model's parameters [37] which in turn also allows for a reduction of the computation time and the memory consumption.

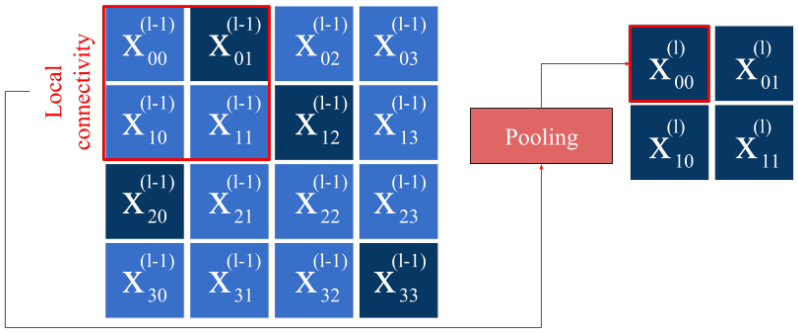


Fig. 3.14 Graphical representation of a 2×2 pooling layer with stride 2. This kernel conducts a sample-based discretization process by which, applying some selection rule given by a numerical operation, it selects the features desired. For instance, if the max pooling is implemented, the dark cells will represent the maximum values within the receptive field of the POOL layer.

As Fig. 3.14 indicates, the POOL layer defines a spatial kernel with size $k \times k$ which is slid over the input volume with a stride $1 < s < k$ in order to reduce the spatial size of each feature map by a s factor. Furthermore, in contrast to the CONV's kernels, the POOL's kernel is not learned, as it defines a specific numerical operation. The most popular operations are the *max-pooling*, which selects the highest value within the window defined by the layer's receptive field, the *sum-pooling*, which sums up all the values within the kernel window, and the *average-pooling*, which in addition to summing up, conducts the average of those values contained in the receptive field [349]. More advanced pooling techniques have also been implemented, such as *mixed pooling* [349], *stochastic pooling* [354], *pyramid pooling*

[140, 351] or *wavelet pooling* [336]. Moreover, some works also propose spectral pooling to summarize data in the spectral domain [280, 305, 359].

Activation function

The FE procedure applied by FC and CONV layers is incomplete if the neuronal unit is not able to distinguish between the most relevant features, i.e those that provide more information to the classification task, and those that are unimportant. In fact, FC and CONV layers are simple linear regressors, as they only apply a linear operation over the data (i.e. the cumulative sum of the matrix multiplication between weights and inputs). In this sense, the activation function plays a key role during the model's learning as it is the principal feature detector mechanism [123]. As we could observe in the previous Eq. (3.9), the mapping function defined by each layer $f^{(l)}$ contains an activation function $\mathscr{H}(\cdot)$, which defines the final output in terms of the combination of those features obtained by the previous FC/CONV layer. Its purpose is to introduce some non-linearity in order to detect (and hence learn) non-linear representations of the data structure, allowing to perform more complex tasks.

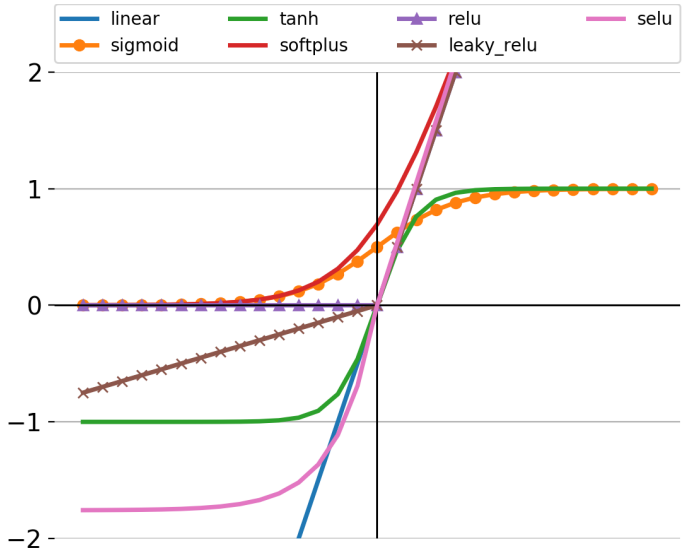


Fig. 3.15 Graphical representation of the most commonly used activation functions within deep neural architectures.

Fig. 3.15 shows some of the most widely used activation functions [6, 225, 251, 269, 307], which should be bounded, continuous, monotonic, and continuously differentiable with respect to the layer's weights [18] with the aim of making learning feasible during backward propagation. Usually, the *sigmoid* $\mathscr{H}(x) = \frac{1}{1+e^{-x}}$ and *tanh* $\mathscr{H}(x) = \frac{e^{x}-e^{-x}}{e+e^{-x}}$ have been widely used in shallow networks with acceptable results. However, their gradient signal tends to fade

when the architectures are too deep (the so-called vanishing gradient problem). To overcome this limitation, new and more effective activation functions have been developed, such as the *rectified linear activation function* (ReLU) $\mathscr{H}(x) = \max(0,x)$ [234] which prevents the vanishing gradient problem at the same time that provides a sparsity function, being more efficient than previous functions, although it can suffer the "dying ReLU" problem [256, 340]. It has inspired other rectified-based functions, such as the *leaky* ReLU (LReLU) [215], the *parametric* ReLU (PReLU) [139] or the *scaled exponential linear unit* (SeLU) [175, 251]. While these functions are generally implemented by hidden layers, there are other interesting functions that are usually applied to the output layers, such as the *softplus* $\mathscr{H}(x) = \ln(1+e^x)$ [91] and *softmax* $\mathscr{H}(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$ activation functions.

3.3.5 Deep neural networks: taxonomy of deep architectures

Deep feedforward neural networks and the design of new layers have been the inspiration for the development of new deep architectures. In this sense, DNNs offer a wide range of deep architectures compared to traditional ANNs thanks to the flexibility of their topologies, connection and types of layers. These deep architectures include autoencoders (AEs), deep belief networks (DBNs), recurrent neural networks (RNNs) and convolutional neural networks (CNNs) as main architectures, with a wide variety of modifications being developed from these models [243]. As it is such a vast field, this thesis reduces its explanation to the basic concepts of the main models.

Stacked autoencoders

AEs are feedforward neural networks composed by input, hidden and output layers (either FC or CONV). Their training is purely unsupervised, as they are focused on representing the input data in a feature space determined by the hidden layer, from which they reconstruct the original input on the output layer. In this sense, if $\mathbf{X} \in \mathscr{X}$ is the original input data in the data space $\mathscr{X}$, the hidden layer maps $\mathbf{X}$ to a code representation $\mathbf{C} \in \mathscr{C}$ by applying its recognition weights $\mathbf{W}_{\mathscr{X}\to\mathscr{C}}$ (also known as encoder components). The reverse process is applied within the output layer, whose generative weights $\mathbf{W}_{\mathscr{C}\to\mathscr{X}}$ (also known as decoder components) map the code dictionary $\mathbf{C}$ to the original representation, obtaining as a result the reconstructed input $\mathbf{X}' \in \mathscr{X}$. This process is described by Eq. (3.19), where $\otimes$ represents the matrix multiplication or the convolution operation, depending on the kind of layer:

$$\mathbf{C} = \mathscr{H}\left(\mathbf{W}_{\mathscr{X}\to\mathscr{C}} \otimes \mathbf{X} + b_{\mathscr{X}\to\mathscr{C}}\right) \tag{3.19a}$$

$$\mathbf{X}' = \mathscr{H}\left(\mathbf{W}_{\mathscr{C}\to\mathscr{X}} \otimes \mathbf{C} + b_{\mathscr{C}\to\mathscr{X}}\right) \tag{3.19b}$$

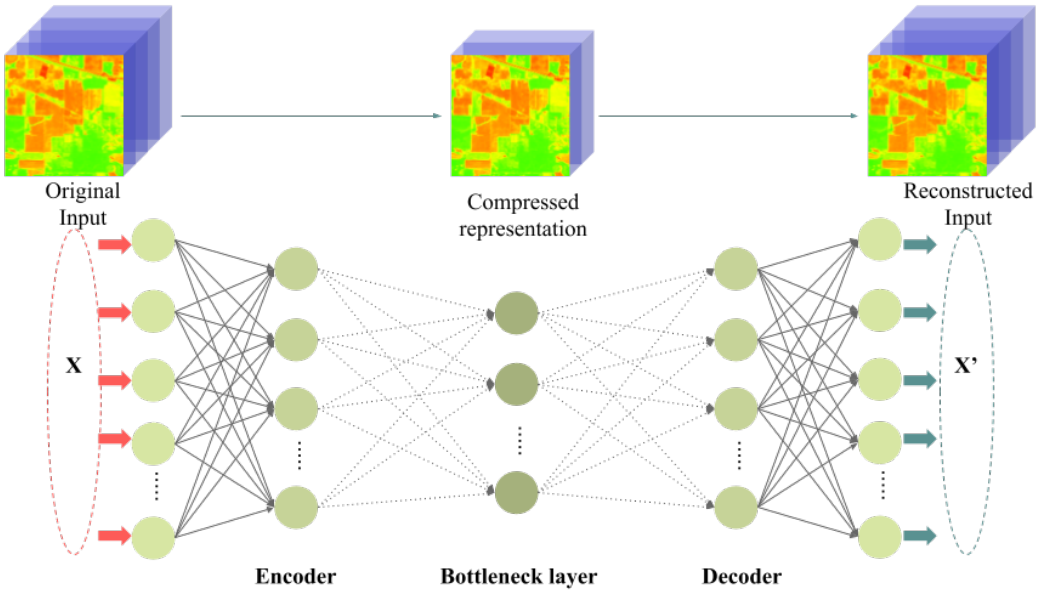


Fig. 3.16 Graphical illustration of a traditional tied SAE for spectral FE. The AE layers on the left-side performs the encoding stage, while the AE layers on the right-side performs the decoding stage. Both stages are connected by a bottleneck layer, that represents the latent space.

Both, $\mathbf{W}_{\mathscr{X}\rightarrow\mathscr{C}}$ and $\mathbf{W}_{\mathscr{C}\rightarrow\mathscr{X}}$ are trained by backpropagation mechanism, where the error signal given by $\mathscr{L} = ||\mathbf{X} - \mathbf{X}'||_2^2$ is propagated backwards through the output and hidden layers.

In this regard, deep (also stacked) AEs (DAEs and/or SAEs) [62] are composed by several concatenated AEs, where the l-th AE corresponds with l-th hidden layer of the model. In this sense, the l-th AE receives as input data the output of the previous AE. Fig. 3.16 represents a traditional tied SAE composed by FC layers, employed for extracting spectral features from HSI data. As it can be observed, several AE layers perform the encoding stage by reducing the spectral dimensionality of the data, until reaching the middle layer of the model (known as bottleneck layer). This layer represents the latent space [19] of the SAE, whose data representations are usually extracted and employed by ML algorithms (such as SVM or LR) to perform the final classification. The following layers perform the decoding stage, until reaching the output layer where the final reconstruction of the input data is obtained.

Deep belief networks

DBNs are composed by several restricted Boltzmann machines (RBMs) [190], which are two-layered ANNs that learn the probability distribution of the input data in an unsupervised

fashion. Focusing on RBMs, they are stochastic generative models where the input layer is composed by *visible* nodes, while the output layer is composed by *hidden* nodes (in fact, there is not a proper output layer, really). In a similar way to AEs, the RBM implements a two-step process. In the first step, the input data is multiplied by the weights that connects the visible and hidden nodes, $\mathbf{W}$, adding a bias b to each hidden node. Then, the obtained result is processed by an activation function, following Eq. (3.20):

$$\mathbf{H}_{(t+1)} = \mathscr{H}\left(\mathbf{V}_t^T \mathbf{W} + b\right) \tag{3.20}$$

where $\mathbf{V}_t$ is the input data at visible units during iteration t, and $\mathbf{H}_{t+1}$ is the resulting data of hidden units. The second step performs is in fact a reconstruction stage, where the reverse process is applied over $\mathbf{H}$ to obtain the desired $\mathbf{V}$, as Eq. (3.21) defines:

$$\mathbf{V}_{t+1} = \mathscr{H}\left(\mathbf{H}_{t+1} \mathbf{W}^T + b\right) \tag{3.21}$$

In this context, the loss function will be defined as $\mathscr{L} = ||\mathbf{V}_t - \mathbf{V}_{t+1}||_2^2$, allowing the unsupervised learning of the RBM's parameters. DBNs takes advantage of these models, concatenating several pre-trained RBMs and refining the full-model's parameters through labeled data.

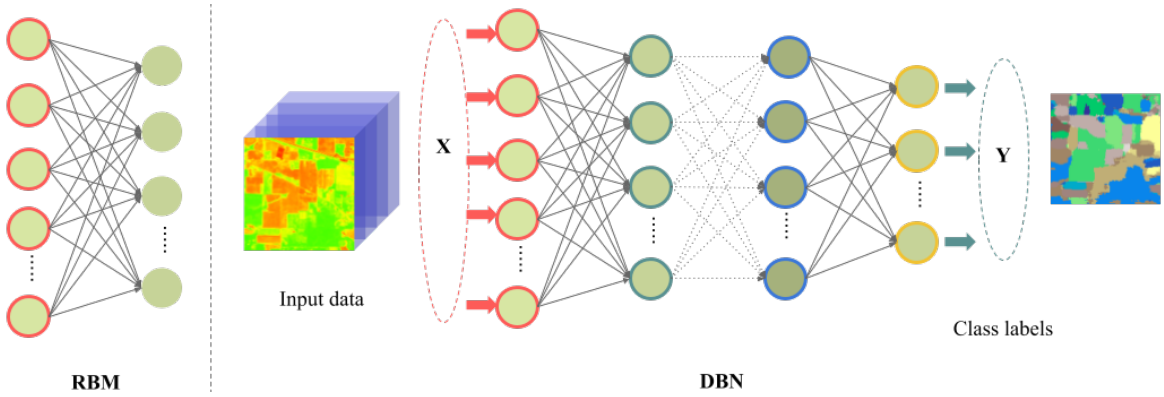


Fig. 3.17 Graphical illustrations of a RBM (left-side) and a DBN (right-side).

Recurrent neural networks

In contrast to feedforward models, RNNs introduce loops in their connections, creating a sequential dependency in which the activations of the nodes in each step depend on those of the previous step. This structure makes the RNN an ideal model for learning temporal sequences, where an internal state or memory allows the association between the current

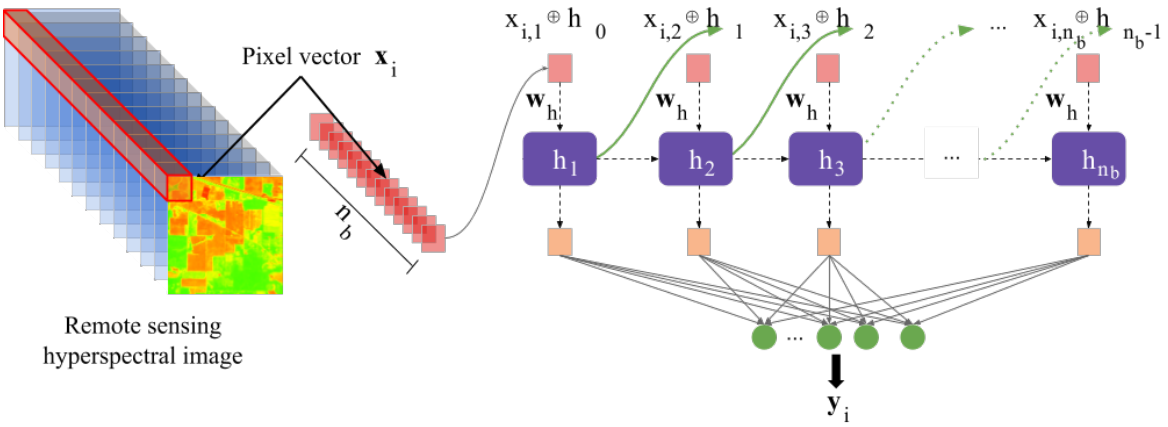


Fig. 3.18 Traditional RNN model for spectral HSI data processing, where the data cube is processed band-by-band within the model. The spectral signature of each pixel is processed as a temporal sequence, obtaining as a result a hidden state **h** that works as the model memory.

input data and the previous ones. This state creates a mechanism for remembering the context of the data, predicting future events depending on those previously remembered.

Depending on the way they create, maintain and update that internal state (see Fig. 3.18), RNNs are classified into three subtypes: vanilla RNNs, which are the most simple ones; long short memory (LSTM) [152], which introduces a complex gate-based mechanism (input, output and forget gates) to control the flow of information, and gated recurrent unit (GRU) [71], which simplifies the gate mechanism, implemented only two gates (updatin and reset gates) to update and clear the model memory. Paoletti et al. [243] provides further details about these models.

Convolutional neural networks

CNNs are deep feedforward models, where the main layer building-block is the CONV layer, which can be combined with non-linear activation and POOL layers to extract, detect and resume the relevant spectral, spatial and spectral-spatial features contained within the data depending on the dimensions of CONV's kernels and the multidimensional input array. In this context, the architecture of any CNN comprises an end-to-end model, in which two parts are clearly distinguished. On the one hand, the FE-net, which is composed by a hierarchical stack of CONV-activation-POOL blocks, which perform the feature extraction and detection stages to learn high-level representations of the input data. On the other hand, the final classifier, which is usually composed by a stack of FC layers, where the output layer provides the final class label. Fig. 3.19 provides a graphical overview about the three basic CNN models that can be applied over HSI data, considering 1D, 2D and 3D kernels.

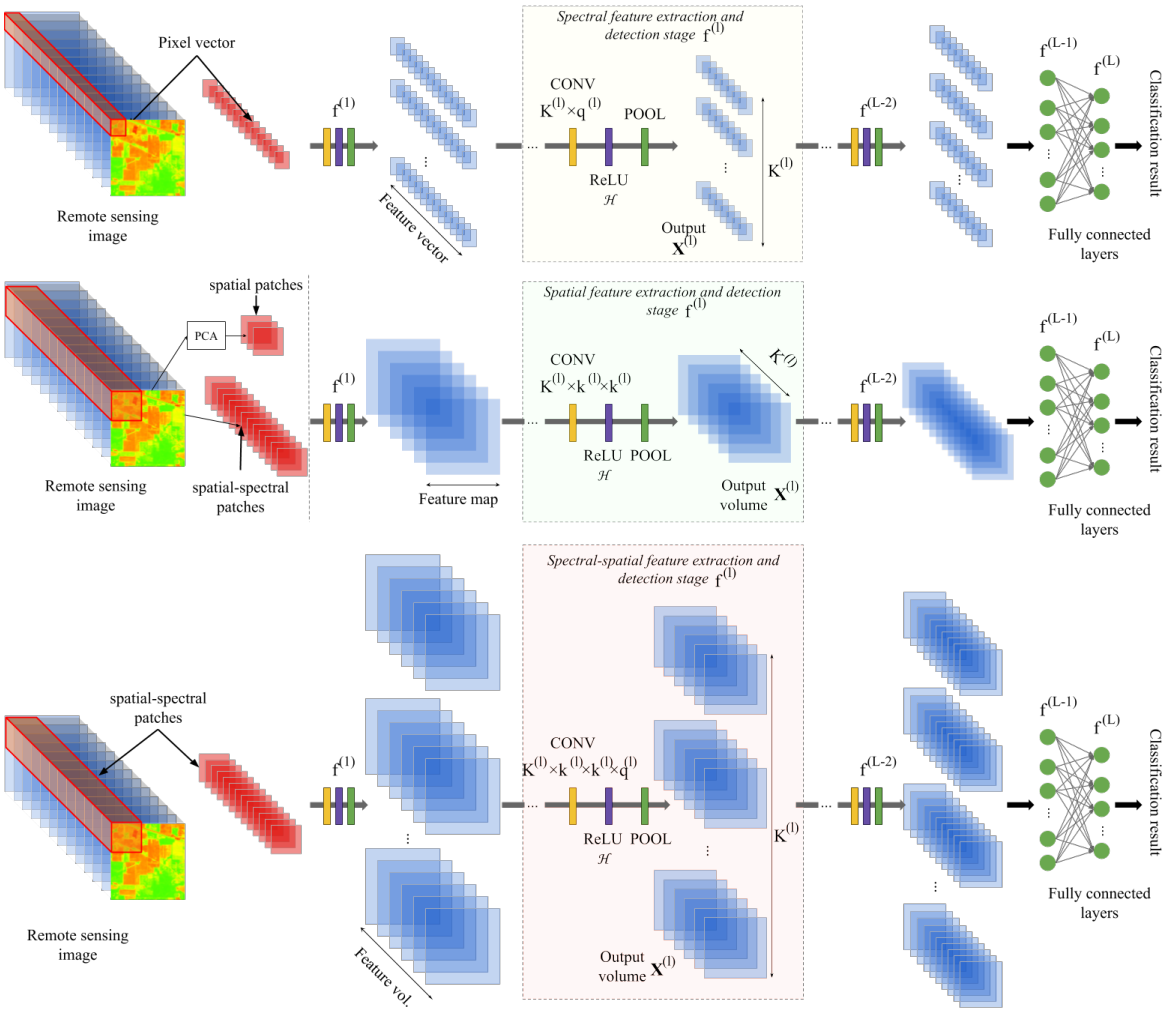


Fig. 3.19 From top to bottom, graphical illustration of CNN1D, CNN2D and CNN3D architectures applied for processing the spectral, spatial and spectral-space information contained in the HSI data cube.

The greatest advantage of CNNs lies in their great potential to extract any kind of feature from the raw data, without applying complex pre-processing mechanisms to it. Furthermore, their flexibility in terms of kernel dimensions, layer connections and model's depth, and the impressive ability to make strong assumptions about the input data have established CNN-based models as the most successful and popular DNN architecture, being extremely popular for HSI data classification. In fact, CNNs represent the current state-of-the-art in image processing, where from time to time, new architectures are developed with the CONV layers as the main inspiration, for instance residual networks (ResNets) [245, 331, 370], dense networks (DenseNets) [242, 333] and generative adversarial networks (GANs) [144, 355].

3.4 Deep neural networks for hyperspectral data analysis

By conducting a detailed review of the existing literature about RS field and DL-based techniques, we can observe a significant increase in the interest that these techniques have inspired in HSI data processing. As a result, a wide variety of works have been designed to exploit deep architectures for spectral, spatial and spectral-spatial classification. In this regard, Fig. 3.20 provides the number of published papers related to DL-based HSI data classification models in the past seven years according to the web of science[5] (WOS).

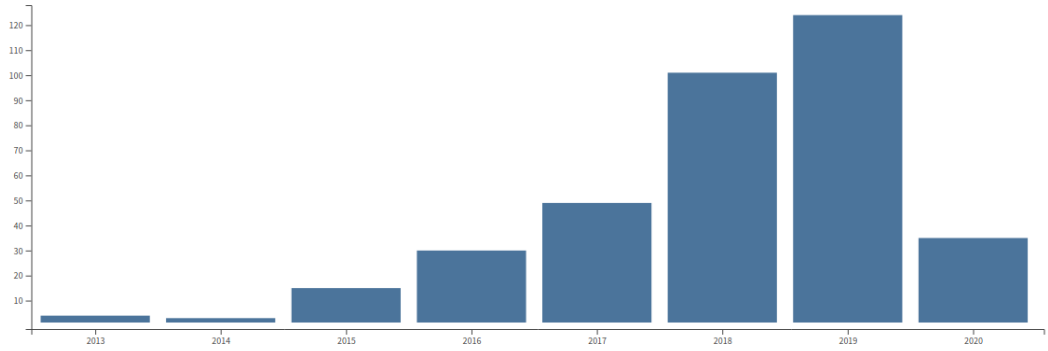


Fig. 3.20 A total of 361 interesting scientific contributions about the processing and classification of HSI with DL-based architectures have been analyzed by WOS and grouped by year (date of acquisition: 14 May 2020).

As we can observe, since the popularization of DL in early 2016 [25], the number of published contributions has been increasing during these years following an upward trend that is expected to continue through 2020. In this regard, the DL for HSI data classification will be further explored, through the implementation of new and more advanced deep architectures in the next several years. Moreover, these contributions are covering a large set of scientific areas and disciplines. For instance, Fig. 3.21 shows the number of contributions organized by area of those articles previously listed by WOS search tool.

Moreover, in the last three years, a large number of literature reviews have been conducted, analyzing the impact these methods have had on the processing of HSI data, not only on the performance of the algorithms in terms of accuracy but also in terms of computational behaviour, such as the contributions of Audebert et al. [14], Gewali et al. [114], Li et al. [202], Liu et al. [211], Narendra and Sivakumar [235], Paoletti et al. [243], Petersson et al.

[5] http://www.webofknowledge.com/WOS
The following search terms have been employed: TS=(hyperspectral AND remote sensing AND classification AND (deep learning OR convolutional OR deep model OR deep feature)) OR TI=(hyperspectral AND remote sensing AND classification AND (deep learning OR convolutional OR deep model OR deep feature))

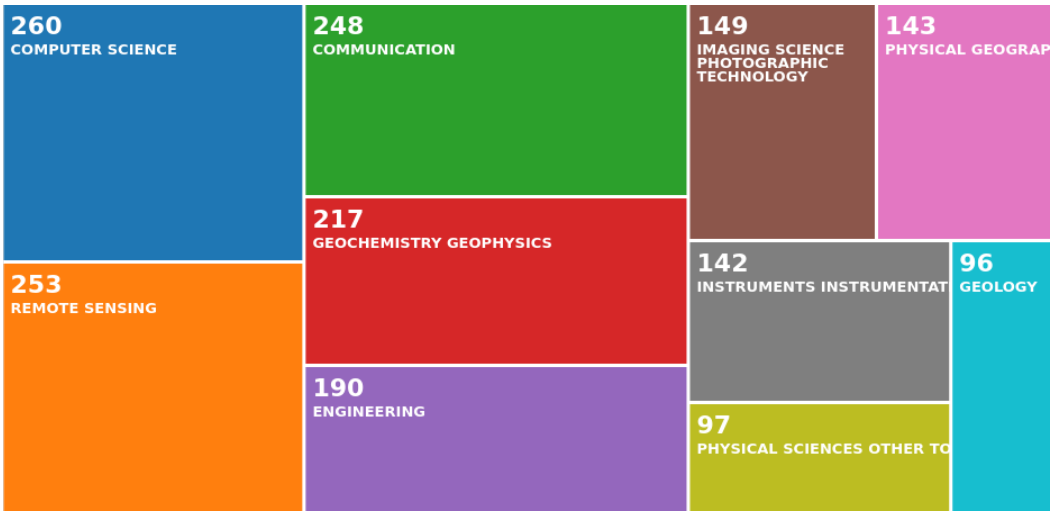


Fig. 3.21 Tree-map by area of those scientific works that are listed by the WOS about DL-based classification methods for RS-HSI data processing (date of acquisition: 14 May 2020).

[260], Signoroni et al. [302], Yuan et al. [350], Zhang et al. [361]. In this sense, Table 3.5 provides some of the most relevant articles about DL-classifiers for HSI data:

Table 3.5 Some relevant contributions about DL models for HSI data classification

Chen et al. [62]	Unsupervised spectral, spatial and spectral-spatial FE by stacked autoencoder (SAE, also known as deep AE, DAE). LR performs the final classification of learned features. KSC and UP scenes are considered for testing purposes.
Wang et al. [328]	Unsupervised spectral FE by stacked denoising AE (SDAE). MLR performs the final classification of those extracted deep-spectral features. Synthetic data and IP scene are considered.
Zabalza et al. [353]	Unsupervised spectral FE by segmented SAE (S-SAE) to reduce the computational complexity of the model. SVM performs the final feature classification. IP image and a subscene extracted from PC (known as Pavia CA) dataset are considered.
Ma et al. [214]	Semisupervised spectral-spatial FE by spatial updated deep AE (SDAE), which is improved by adding a regularization term to determine the spatial similarity of the samples. IP, PC and Botswana scenes are considered for testing purposes.

Chen et al. [63]	Unsupervised spectral, spatial and spectral-spatial FE by deep belief network (DBN). LR performs the final feature classification. IP and UP scenes are considered for testing purposes.
Zhong et al. [368]	Unsupervised spectral FE by DBN, which is eventually fine-tuned in a supervised way by backpropagation for classification purposes. IP and UP scenes are considered for testing purposes.
Mou et al. [229]	Spectral classification by RNN, implementing a modified gated recurrent unit (GRU) and adopting the parametric rectified tanh (PRetanh) activation function. IP, UP and UH scenes are considered for testing purposes.
Zhou et al. [371]	Spectral, spatial and spectral-spatial HSI data classification by RNN, implementing a long short term memory (LSTM) based architecture. IP, UP and KSC scenes are considered for testing purposes.
Hu et al. [154]	Spectral HSI data classification by CNN1D model. IP, SV and UP scenes are considered for testing purposes.
Chen et al. [61]	Spectral, spatial and spectral-spatial HSI data classification by CNN1D, CNN2D and CNN3D models with LR. IP, UP and KSC scenes are considered for testing purposes.
Liang and Li [205]	Spatial FE with CNN2D, which are enhanced through SC. IP and UP have been considered for testing purposes.

The success of these deep models within HSI data classification tasks lies in their "Swiss army knife" behaviour due to its impressive flexibility to process the spectral, spatial and spectral-spatial information contained into the HSI data cube in a natural and simple way, where the same model is able to process these types of features with any changes or adaptations in its structure, such as SAEs and CNNs, where the spectral-spatial features can be concatenated as a 1D-array or by selecting the suitable kernel dimension. Moreover, their hierarchical structure allows to extract hidden and sophisticated patterns and structures within the raw HSI data with different levels of abstraction (low-level at the bottom layers and high-level at the top layers), which in turn allows us to choose the appropriate "degree of depth" to solve the classification problem, that is, for simple datasets shallow models will be enough, while for larger and more complex datasets, deeper networks can be considered. As a result, we can easily adapt the model to the complexity of the problem, leading to precision results never seen before, thanks to which DNNs are considered as the current state-of-the-art in image processing [141]. In addition to the flexibility in terms of the model's depth and the kind of features it can extract (i.e. the wide variety of architectures and neural layers

that are available), the ability to generalise any type of knowledge using almost any learning approach allows us to address the classification problem using different strategies, employing a wide range of methodologies, from unsupervised models as SAEs and DBNs to supervised ones, like RNNs and CNNs, being able to apply intermediate solutions belonging to the wide field of semi-supervised learning [97, 136, 198, 211, 338]

These features make DNNs very powerful models for HSI data processing. However, like almost all ML algorithms, the development of these models has certain intrinsic limitations that may hinder their performance. Moreover, these limitations can be really exacerbated when dealing with the challenges imposed by HSI data.

3.4.1 Challenges and limitations of deep models when facing hyperspectral data classification

As noted above, DNN models face some challenges related to processing high-dimensional data sets, such as HSI data cubes. In fact, any ML/DL method has to take into account that, despite the strong conviction that the spectral resolution of HSI data should enhance the separability between different land cover classes, it actually introduces more complexity to processing models, even having adverse effects on their performance. This adverse behavior observed within the algorithms when facing high-dimensional data has been mathematically demonstrated as the *peaking paradox* [168, 303, 319]. The logic behind this paradox is quite intuitive: as the feature domain grows, the number of spectral bands $\mathbf{x}_i \in \mathbb{N}^{n_b} = [x_{1,i}, \ldots, x_{i,n_b}]$ to be considered by the processing method increases, so the number of statistical parameters to cover that information will be increased too. More parameters means that the method will have to adjust more degrees of freedom, so the estimation errors will boost, hampering the performance [189]. Nevertheless, the peaking paradox is only the tip of the iceberg, leading to the *curse of the dimensionality* of HSI data classification methods [23]. In this regard, classification methods need more training data due to the high number of statistical parameters that must be correctly estimated. However, beyond a critical spectral size, the training set does not contain enough information to properly refine and adjust the large amount of parameters. As a result, the models are poorly adjusted to the data, producing over-fitting problems. In this sense, despite achieving acceptable accuracy during the training stage, the model is not able to generalize the knowledge to new samples, so the accuracy during inference is usually quite poor (*Hughes phenomenon* [158]).

In addition to the negative performance of classifiers when dealing with high-dimensional feature space, the internal characteristics of these data must be taken into account, too. In this sense, HSI data introduces several artifacts that make the classification process harder.

On the one hand, uncontrolled changes in the reflectance $\rho(\lambda)$ captured by the spectrometer at each wavelength λ can introduce a great intra-class variance within the spectral signatures of the HSI data cube. These are usually produced by changes in atmospheric conditions that introduces variations in the scene illumination. On the other hand, instrument noise can degrade the collected information, hindering the quality of the data cube collected by corrupting the spectral signatures [273], even preventing several bands from being used due to saturation/cutoff or calibration errors [255]. Furthermore, since the spectrometer measures the solar radiation by scattering the light beam that passes through an entrance slit by means of a refracting element, some blurring and smoothing effects can be introduced due to the width of the slit. Also, other factors, such as a poor radiometric accuracy and SNR, can impact negatively on the quality of the captured spectral data. Regarding the spatial information, HSI pixels often cover large regions on the surface of the Earth due to the trade-off between spectral bandwidth and spatial resolution. As a result, HSI data cubes have low/medium spatial resolution, where the pixels tend to exhibit mixed spectral signatures, leading to high inter-class similarity in border regions. In the end, these challenges create potential ambiguities and uncertainties [121, 325].

Besides these issues, the problematic lack of labeled samples prevents the correct training of deeper and more complex models, exacerbating the curse of the dimensionality, in the sense that there are not enough samples to cover the high data variability. This is due to several factors: on the one hand, despite the launch of new HSI-EO missions (such as the EnMAP or PRISMA projects), satellite HSI-based observation missions remain poorly represented within EO field due to their technical constraints and practical challenges. As a result, the number of current operational spectrometers are still low compared to other types of remote sensors, such as those spatial-high-resolution and MSI devices in Landsat, Sentinel and SPOT systems. On the other hand, the airborne spectrometers cover much smaller areas than the satellite-based sensors, so the amount of HSI data sets is limited. In addition, repositories with labeled HSI are usually not openly and publicly available, being the tagging of each pixel and arduous, time-consuming and expensive task as it requires a human expert.

In addition to the challenges introduced by HSI data, the inherent limitations of current DL-based models should be considered [238]. In this context, while HSI data includes an important degree of difficulty compared to other types of RS data because of its high dimensionality, DL involves a greater complexity in the classifiers, by increasing the number of parameters required by deep models. Consider the training stage, for instance, the learning and adjustment of a DNN's parameters is formulated as a non-convex and NP-complete problem [36], which must be solved through an optimization process without guarantee of

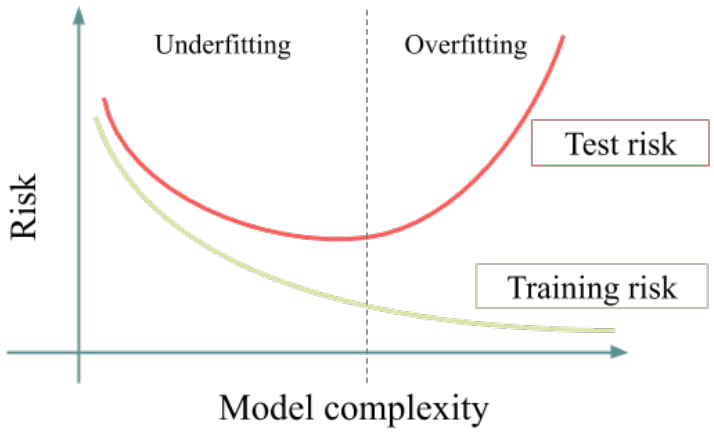


Fig. 3.22 Curves for test and training risks when compared to the complexity of the deep model, i.e. the number of parameters employed by the model. From left to right, given a training set $\mathscr{D}_{train}$, as the model includes more parameters, the training error decreases until be close to zero, by adjusting their degrees of freedom to the training data. However, the test error begins to grow after a number of parameters, because the network is not able to generalize the data distribution, so it cannot apply its knowledge about unseen data..

convergence, since by introducing more learnable parameters more degrees of freedom are considered that can lead to multiple local minima [17], where the optimizer (and therefore the model's learning) gets stuck [60, 237].

The increase in depth, and therefore in the number of parameters, implies a very high consumption of training data in order to learn properly the model's weights and biases. Thus, DNNs tend to overfit (see Fig. 3.22) when there are few training parameters [96]. In this regard, the high-dimensional nature of HSI data, its high variability and the limited availability of training samples greatly hamper the generalization ability of deep models. Furthermore, as the depth of the model increases, the forward propagation suffers a significant degradation [141], while the backpropagation suffers some difficulties in propagating the gradient signal to all layers [310]. In fact, the gradient fades slightly as it goes through each layer, resulting in its practical vanishing when the depth is extremely large. These problems elongate the loss function $\mathscr{L}$ until the model cannot properly adjust its parameters at each iteration. In addition, DL-based models run much slower than classical ML algorithms due to the large amount of parameters that must be managed, requiring computationally expensive and memory-intensive methods [67]. In this sense, a high computational burden is involved due to the large amount of parameters that must be managed, therefore DNNs often require hardware accelerators (such as GPUs) in order to reduce computation times (in fact, to make the training feasible).

Finally, deep models implements a "black box" mechanism during the training stage, where the internal models' dynamics are very hard to interpret [27, 208]. In fact, designers have no control over the features that the model will extract in each layer, which may hinder the design and implementation of optimization decisions.

3.5 Contributions of this thesis to the scientific field

This thesis focuses its efforts on providing some solutions to the the above-mentioned limitations that DL models suffer when face the HSI data classification problem. In this sense, the main goal is the design, implementation and validation of new models and methodologies based on deep architectures that not only improve the accuracy results of the classifiers, but also their computational performance in terms of run times and memory consumption.

With this purpose in mind, different tasks have been conducted throughout the development of this thesis. The first one has consisted in the analysis of the current state-of-the-art about HSI data within EO and RS fields, exploring in detail the characteristics of current operational sensors and the available data repositories, identifying the main characteristics, challenges, difficulties and limitations of the available data[6]. This analysis has focused in particular on the classification techniques used on HSI data.

After understanding the idiosyncrasies of HSI data, the second task has reviewed in the available literature the current developed DL architectures, understanding its evolution from the traditional shallow ANN models to the current models with an impressive depth and complexity, and composed of different types of layers. This has be done by listing their advantages and limitations in comparison to traditional HSI-ML methods, and exploring their adaptation and application to HSI data classification. In this way, the main DL models used have been identified, as well as the way in which they are used and the results obtained in comparison with traditional ML methods[7].

Once strengths and weaknesses of deep models have been identified when solving HSI data classification problems, the third task is to design, implement and validate new methodologies and architectures that overcome the limitations of current models, with the aim of providing a faster, more accurate and more efficient analysis of the spectral-spatial information contained in the HSI data cubes. In this sense, new DL-based architectures have been implemented for HSI data classification, developing distributed many-core solutions based on GPU devices and conducting an exhaustive comparative analysis of the improvements developed in real applications, using the real HSI images described in section 3.2.2.

[6]The knowledge acquired has been summarized in sections 3.1, 3.2.2 and 3.4.1.

[7]The knowledge acquired has been summarized in sections 3.2, 3.3 and 3.4.

In this context, proposed methods have been evaluated in terms of accuracy performance, considering some well-known metrics such as overall accuracy (OA), average accuray (AA) and kappa coefficient (K). The complexity of models in terms of number of parameters and run times has also been taken into account to measure their computational performance, conducting a comprehensive and fair comparison with the classification methods employed by the scientific community.

In this regard, following sections will summarize each of the contributions included in this publication compendium. Given the number of different models implemented, they will be presented one by one, from the simplest to the most complex one.

3.5.1 Review about deep hyperspectral data classifiers

Although it is not an article that presents new implementations or methodological solutions to the scientific field, the work entitled *Deep Learning Classifiers for Hyperspectral Imaging: A Review* [243] has been included in this compendium due to the extensive overview about DL for HSI data classification it presents. In this sense, it is a very interesting contribution to delve into the concepts that this thesis has been introducing within the previous sections, in a clear and contextualised way. Thus, it presents a wide variety of architectures for spectral, spatial and spectral-spatial HSI analysis based on SAEs, DBNs, RNNs and CNNs models, highlighting the new trends that the scientific community is developing to improve classification results, such as ResNets [245, 331, 370] and DenseNets [242, 333]. Moreover, it conducts a detailed comparison between a large set of DL models (such as MLP, vanilla, GRU and LSTM RNNs, CNN1D, CNN2D and CNN3D) and available publications [115, 136, 171, 242, 244, 245, 370], considering also different amounts of spectral-spatial information.

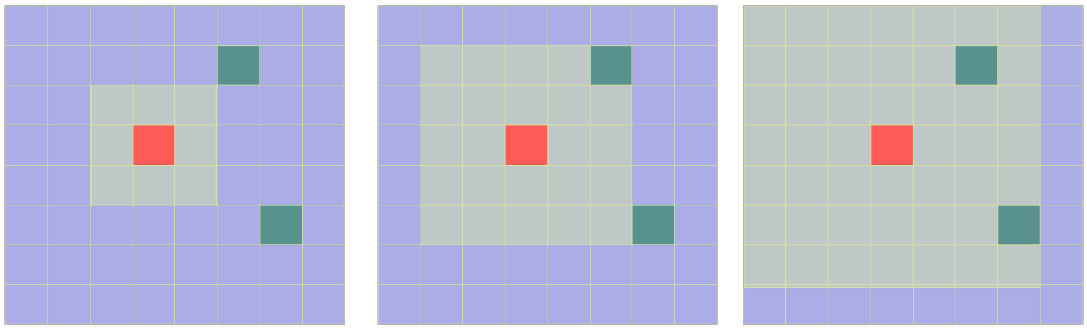


Fig. 3.23 Graphical representation of the spatial overlapping between the current training sample (in red) and the test samples (in green) considering different neighbourhood window sizes for the training sample, i.e. 3×3, 5×5 and 7×7.

In addition, this review is one of the first that introduces the controversy about the use of spatial information in supervised classifiers when random selection mechanism selects the training samples, which can be spatially overlapped with the test samples as described in Fig. 3.23, an undesired effect that could significantly facilitate the subsequent classification of the test samples during the inference stage (as they have been previously processed in some way by the model during the training step). All the experiments were conducted over IP, UP, SV and UH datasets, considering also the spatially disjoint samples of IP and UP datasets.

3.5.2 Improving the scalability of recurrent neural models

As pointed before, the RNN is composed by recurrent cells with internal loops between their connections, allowing to keep the learned information within the model during some iterations. In this sense, the spectral signatures contained into the HSI data cube can be considered as a time sequence in which, at each timestep, the spectral value changes, existing a dependency between the previous and subsequent bands [229].

The RNN model has been applied for data classification through three main architectures, which depends on the desing of their recurrent cell. The vanilla recurrent unit (VRU) is the simplest unit, which hampers its ability to discover hidden relationships and patterns within complex sequences like spectral signatures. The LSTM obtains better accuracy than the previous one by including three gates to control the information flow. However, due to the complexity of the control-gate mechanism, deep recurrent architectures could be affected by the vanishing gradient problem. Finally, the GRU tries to avoid the previous limitation by simplifying the internal structure of the cell, seeking a balance between the VRU and LST to minimize the impact of the vanishing gradient problem.

Focusing on HSI data classification, these models can achieve acceptable accuracy considering that they only take into account spectral information. However, they scales poorly when dealing with high-dimensional data cubes, due to the high dependence on previous steps. Although the vast majority of publications have optimized and parallelized the algebraic operations, the dependencies between the current and old hidden states hampering the speed up of these models, badly affecting the performance when the dimensionality and the amount of data grow. In order to mitigate this problem, this thesis incorporates the proposal for improvement presented in the entitled work *Scalable Recurrent Neural Network for Hyperspectral Image Classification* [249]. This paper presents a new RNN classifier based on simple recurrent units that performs HSI classification in a highly scalable and efficient way. In particular, it adopts the simple recurrent unit (SRU) [197] as the recurrent cell body, which maintains two internal states, the hidden and the cell state, controlled by two gates (forgot and reset gates). Moreover, the entire network has been parallelized on a GPU

device in order to obtain not only good accuracy results but to achieve also a competitive performance in terms of run times and data scalability. The quantitative and qualitative results obtained from the experiments conducted over the AVIRIS' IP, BIP and SV scenes reveal an interesting performance, not only in terms of classification accuracy (in line with existing methods), but also in terms of computational performance when dealing with large datasets.

3.5.3 Enhancing the convolutional neural network for fast and accurate hyperspectral data classification considering spectral-spatial features

As pointed in section 3.4.1, deep networks in general, and CNNs in particular exhibit a high computational burden due to the large number of parameters that must be trained. It should be noted that each CONV layer with kernel 3D involves $((k^{(l)} \cdot k^{(l)} \cdot q^{(l)}) + 1) \cdot K^{(l)}$ 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. This is worsened by the high dimensionality of the HSI data. As a result, the first CNN models for processing the spectral-spatial information contained into the HSI data cube exhibited very high run times, so they often applied PCA to reduce the spectral information. For instance Chen et al. [61] summarizes the spectral information in 32 bands.

In this context, the work *A New Deep Convolutional Neural Network for Fast Hyperspectral Image Classification* [246] proposes a CNN architecture to process the spectral-spatial features from the raw HSI data cube, without summarizing the spectral information. The architecture has been efficiently implemented on a GPU device to parallelize the algebraic operations required during kernel computation. In addition, this work conducts a detailed study of how spatial information affects the classifier by testing different input spatial sizes. Experiments have been conducted over IP and UP scenes, reaching competitive results in comparison with other classifiers such as the SVM or the CNN developed by Chen et al. [61].

3.5.4 Enhancing the feature extraction during the training stage

CNN models suffer a major problem of over-fitting when few training samples are available. To avoid this limitation, some works propose the use of regularization techniques that improve the training of the model, by extracting more robust, discriminative and decoupled features from the original data. In this context, dropout regularization [151, 309] has proven to be a good solution to enhance the performance and robustness of the CNN model. It is able to

avoid complex co-adaptations on training data by randomly deactivating a percentage of the neural activations. As a result, the generalization ability of the model is drastically enhanced. However, although traditional regularization dropout strategy has been shown to be effective in FC-based architectures, it does not achieve the desired performance in the CONV layers, as it maintains the spatial dependencies in the resulting feature maps.

To overcoming this limitation, this thesis develops the work entitled *Neighboring Region Dropout for Hyperspectral Image Classification* [247], which introduces the neighboring region dropout technique to selectively cuts off certain neighboring outputs, creating spatial dropped regions within the CONV layer's feature maps. This technique has been tested over IP and UP datasets. The obtained experimental results reveal that the newly proposed method helps to achieve better classification accuracy than traditional dropout strategy, even when very few training samples are available, with a low computational cost.

3.5.5 Avoiding vanishing gradient problem in deep convolutional neural networks through attention mechanism

To overcome the data degradation during the forward pass and the the vanishing of the gradient during the backpropagation, some works propose the implementation of short paths from low-level layers to high-level layers, i.e. *residual connections* [141]. As we can observe in Fig. 3.24, several FE and detection stages (i.e. CONV, activation and POOL layers) are grouped into residual units, whose inputs are directly connected to their outputs through an aggregation operation. In this sense, the residual connection applies an identity mapping that helps to propagate previous information to the subsequent units, alleviating the forward data degradation and improving the backward step by promoting the propagation of the gradient.

Inspired by the residual learning, this thesis proposes the work entitled *Visual Attention-Driven Hyperspectral Image Classification* [138] with the aim of improving the performance of CNNs and ResNets through attention techniques. The implemented network takes advantage of residual and skip connections along the CNN model, applying a new visual attention mechanism to increase the sensitivity of the network to those features of the HSI scene that contain the most important and useful information for classification purposes. This is done by applying masks that have been automatically learned during the training stage of the model. The basic idea is to "open" the black box of the CNN model, implementing a training guided by the most relevant features. The proposal was tested over IP, UP, SV and UH datasets. The obtained results demonstrate that the approach is more robust to over-fitting than the current state-of-the-art methods, achieving higher accuracy values with very small training sets and exhibiting a greater tolerance to data disturbances.

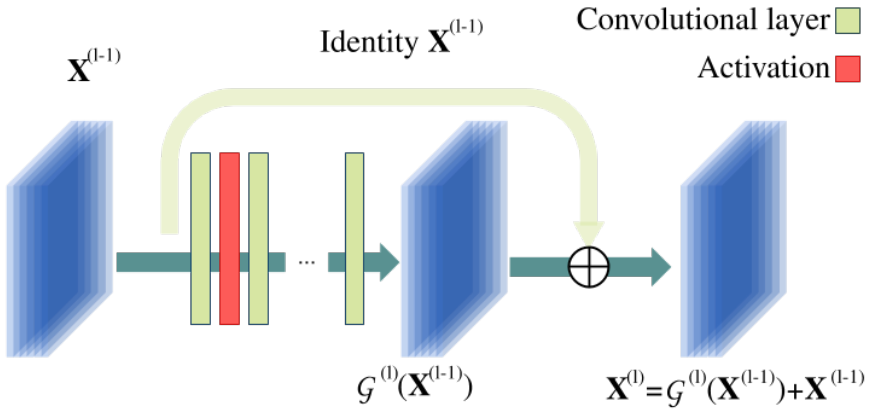


Fig. 3.24 Graphical representation of the residual unit, where the inputs and outputs of a block of layers are combined through an aggregation function. $\mathscr{G}^{(l)}$ defines all the mapping functions performed within the residual block.

3.5.6 Reducing the number of parameters through a continuous interpretation of residual models

A very interesting approach to alleviate the over-fitting of deep models is to reduce their complexity in terms of the number of parameters to be trained. In this sense, it should be noted that deep models may not only incur serious over-fitting problems, but also excessive memory consumption. This is worsened in those architectures such as the ResNet, whose memory consumption increases directly with each residual block introduced into the model. Therefore, the thinning of the network make the deep model lighter, reducing the number of parameters which leads to faster training and execution and lower memory consumption.

To overcome this limitation, this thesis presents the work *Neural Ordinary Differential Equations for Hyperspectral Image Classification* [248], which re-interprets the traditional-discrete ResNet as a continuous-time ordinary differential equation (ODE), as Fig. 3.25 indicates. The ODE function is designed as a CNN and combined with a deep architecture. As a result, the implemented model drastically reduces the number of parameters (and therefore the memory consumption) to be trained, offering a great flexibility when processing and classifying HSI data. The model has been developed on a GPU devices, being tested over IP, UP, SV and KSC datasets. A significant performance has been achieved even when a very few training samples are available in comparison with other ResNet implementations, so it proves to be a robust solution to over-fitting problem.

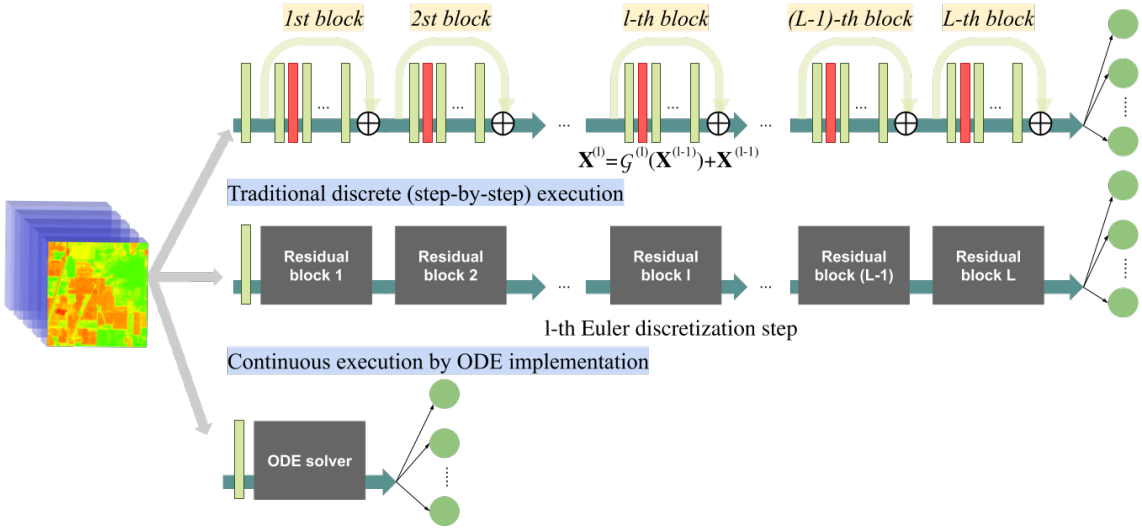


Fig. 3.25 Graphical representation of the residual unit when it is considered discreetly as a set of residual blocks and continuously through an ODE.

3.5.7 Developing new deep architectures for accurate hyperspectral data classification

In addition to those limitations described in section 3.4.1, convolution-based architectures have a limited capacity to exploit the spatial relationships between features detected at different positions within the image. This is due to the spatial summary that the POOL layers make in the extracted feature maps. In fact, although the POOL layer contributes in a positive way by providing certain invariance to data translations, its use implies a certain loss of spatial information by disregarding how different features are spatially related to each others.

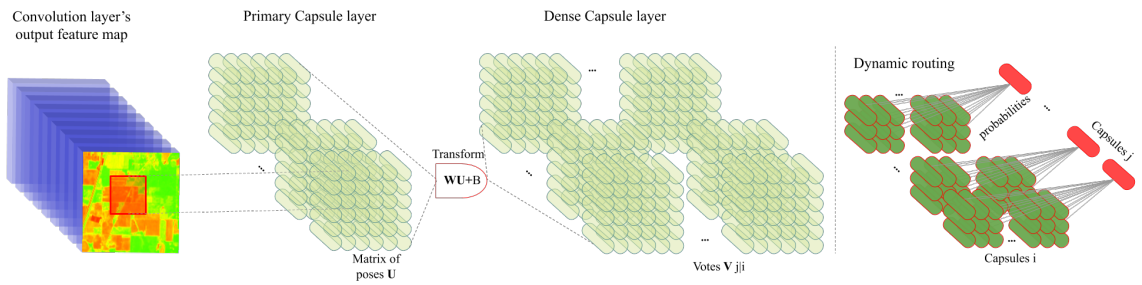


Fig. 3.26 Graphical visualization of the HSI-CapsNet, where each capsule unit is composed by a set of convolution layers. Communication between the capsules belonging to different layers is done by dynamic routing which calculates the probability that the capsules in the lower layers can activate the capsules in the upper layers.

As an improvement of the convolutional architecture, this thesis proposed the adoption of capsule layers [287] (see Fig. 3.26) for the first time in the literature with the aim of performing spectral-spatial HSI data processing. This architecture encodes the data internal relationships into an activity vector instead of the traditional scalar value of the convolution feature map. Such data representation has demonstrated to be powerful in encoding useful features from the data, solving the limitations exhibited by the pooling layer. In this sense, the entitled work *Capsule Networks for Hyperspectral Image Classification* [244] develops and tests the spectral-spatial CapsNet for HSI classification over a wide variety of HSI images, which includes information about agricultural and urban areas. In particular, IP, UP, SV, KSC and BIP scenes have been considered, outperforming the accuracies reached by the current state-of-the-art, which is mainly composed by traditional CNN models and ResNets.

Conclusions

DL methods have revolutionized image analysis and proved to be a powerful tool for processing high-dimensional remotely sensed data. Hierarchical structures, composed by chains of operational layers/blocks concatenated one after another, demonstrate a great generalization power, working as universal approximators. In particular, their behavior can be adapted to the special characteristics of HSI data in order to perform classification tasks. This thesis has conducted an exhaustive analysis of the most popular DL models for the HSI processing, where those CNN-based architectures have been found to be particularly effective, due to their ability to extract highly discriminatory features and effectively leverage the spatial-contextual and spectral information contained in HSI data cubes. In this sense, the challenges of both deep models and HSI data have been analysed, exploring also the challenges of HSI classification with DL techniques.

Special attention has been paid to new trends in the current state-of-the-art, reviewing the strengths and weaknesses of the most widely used techniques for improving the performance of DL architectures. In partcicular, the performance of the DL models can be significantly improved through through new connections (both, residual and skip connections) and data paths. Moreover, the use of standardization techniques, together with the reusability of the information contained in HSI data via residual connections and the concatenation of different paths have allowed to overcome important problems such as over-fitting and the vanishing gradient when few training samples are available, or when very deep structures are implemented.

One of the main aspects preventing the full adaptation of the discussed paradigms to practical problems is that most of the considered models are highly demanding in computational terms, particularly when applied to complex HSI scenes. However, advances in computer technology and hardware platforms are rapidly allowing to increment the complexity and depth of the networks, making the required fine-tuning processes feasible in a reasonable amount of time. In this sense, there have been several efforts in the field of hardware accelerators that have made possible to implement deep models into embedded processors, such as GPUs, which can effectively parallelize the workload of DL-based net-

works [270, 86, 366, 133]. Regarding to this, the optimization and parallelization of these models is an attractive research direction which can provide efficient mechanisms to address the enormous computational requirements introduced by DL-based HSI data processing, since the acquisition ratios of imaging spectrometers and and the volume of future available repositories are expected to be extremely large [32].

In addition to the comprehensive analysis of HSI-DL, this thesis provides six new works to the scientific community that have been published in the journals in the area of remote sensing and computer science, all of them with impact factor. These publications provide significant novelties both in the developing of new architectures for data processing (such as [248, 138] and [244]), and in the implementation of additional techniques for enhancing the performance of standard models (such as [247]), also providing several model optimizations through an efficient data management and the parallelization over GPU devices (such as [249, 246]).

In addition to these outstanding works, this thesis has been very productive in terms of scientific contributions to the HSI remote sensing community, contributing with 28 scientific journal articles. Of these works, some have been distinguished with certain merits for their impact on the scientific community. For instance, four of them have been labeled as "Highly Cited Papers" in Web of Science/Clarivate Analytics Essential Science Indicators (ESI), being the publication *A new deep convolutional neural network for fast hyperspectral image classification* [246] considered as hot paper during its first two years and the Most Cited ISPRS Journal of Photogrammetry and Remote Sensing Articles since 2017[8]. Moreover, one of the most recent articles [250] has been selected as the cover of the Multidisciplinary Digital Publishing Institute (MDPI) Remote Sensing journal of april 2020. Their contributions in congresses have also been productive, being one of them [216] the recipient of the best paper award of the IEEE Workshop on Hyperspectral Image Processing 2019 (WHISPERSS'19) congress.

Finally, the PhD student has been recipient of the recognition of Best Reviewers of the IEEE Geoscience and Remote Sensing Letters journal in 2020, and she is currently editing three special issues in the MDPI Remote sensing that are closely related to the topic of the thesis:

- Remote Sensing: "Big Data in Earth Observation: A New Computing Paradigm for Remote Data Analysis".
- Remote Sensing: "Deep Neural Networks for Remote Sensing Applications".

[8]Information available on
https://www.journals.elsevier.com/isprs-journal-of-photogrammetry-and-remote-sensing/most-cited-articles

- Remote Sensing: "Convolutional Neural Networks for Object Detection".

References

[1] Ablin, R. and Sulochana, C. H. (2013). A survey of hyperspectral image classification in remote sensing. *International Journal of Advanced Research in Computer and Communication Engineering*, 2(8):2986–3000.

[2] Acosta, I. C. C., Khodadadzadeh, M., Tusa, L., Ghamisi, P., and Gloaguen, R. (2019). A machine learning framework for drill-core mineral mapping using hyperspectral and high-resolution mineralogical data fusion. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*.

[3] Adam, E., Mutanga, O., and Rugege, D. (2010). Multispectral and hyperspectral remote sensing for identification and mapping of wetland vegetation: a review. *Wetlands Ecology and Management*, 18(3):281–296.

[4] Adão, T., Hruška, J., Pádua, L., Bessa, J., Peres, E., Morais, R., and Sousa, J. J. (2017). Hyperspectral imaging: A review on uav-based sensors, data processing and applications for agriculture and forestry. *Remote Sensing*, 9(11):1110.

[5] Aggarwal, S. (2004). Principles of remote sensing. *Satellite remote sensing and GIS applications in agricultural meteorology*, pages 23–38.

[6] Agostinelli, F., Hoffman, M. D., Sadowski, P. J., and Baldi, P. (2014). Learning activation functions to improve deep neural networks. *arXiv preprint arXiv:1412.6830*.

[7] Al-khafaji, S. L., Zhou, J., Zia, A., and Liew, A. W. (2018). Spectral-spatial scale invariant feature transform for hyperspectral images. *IEEE Transactions on Image Processing*, 27(2):837–850.

[8] Alcolea, A., Paoletti, M. E., Haut, J. M., Resano, J., and Plaza, A. (2020). Inference in supervised spectral classifiers for on-board hyperspectral imaging: An overview. *Remote Sensing*, 12(3):534.

[9] Andrews, R., Diederich, J., and Tickle, A. B. (1995). Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-based systems*, 8(6):373–389.

[10] Arablouei, R., Goan, E., Gensemer, S., and Kusy, B. (2016). Fast and robust pushbroom hyperspectral imaging via dmd-based scanning. In *Novel Optical Systems Design and Optimization XIX*, volume 9948, page 99480A. International Society for Optics and Photonics.

[11] Arsenault, É. and Bonn, F. (2005). Evaluation of soil erosion protective cover by crop residues using vegetation indices and spectral mixture analysis of multispectral and hyperspectral data. *Catena*, 62(2-3):157–172.

[12] Aspinall, R. J. (2002). Use of logistic regression for validation of maps of the spatial distribution of vegetation species derived from high spatial resolution hyperspectral remotely sensed data. *Ecological Modelling*, 157(2-3):301–312.

[13] Asrar, G. (1989). *Theory and Applications of Optical Remote Sensing*. Wiley Series in Remote Sensing and Image Processing. Wiley.

[14] Audebert, N., Le Saux, B., and Lefèvre, S. (2019). Deep learning for classification of hyperspectral data: A comparative review. *IEEE Geoscience and Remote Sensing Magazine*, 7(2):159–173.

[15] Azam, S. and Gavrilova, M. L. (2017). Biometric pattern recognition from social media aesthetics. *International Journal of Cognitive Informatics and Natural Intelligence (IJCINI)*, 11(3):1–16.

[16] Babey, S. and Anger, C. (1989). A compact airborne spectrographic imager (casi). In *Quantitative Remote Sensing: An Economic Tool for the Nineties, Volume 1*, pages 1028–1031.

[17] Bach, F. (2017). Breaking the curse of dimensionality with convex neural networks. *Journal of Machine Learning Research*, 18(19):1–53.

[18] Bakr, M. H. and Negm, M. H. (2012). Modeling and design of high-frequency structures using artificial neural networks and space mapping. In *Advances in Imaging and Electron Physics*, volume 174, pages 223–260. Elsevier.

[19] Baldi, P. and Hornik, K. (1989). Neural networks and principal component analysis: Learning from examples without local minima. *Neural networks*, 2(1):53–58.

[20] Bannari, A., Pacheco, A., Staenz, K., McNairn, H., and Omari, K. (2006). Estimating and mapping crop residues cover on agricultural lands using hyperspectral and ikonos data. *Remote Sensing of Environment*, 104(4):447 – 459.

[21] Bareth, G., Aasen, H., Bendig, J., Gnyp, M. L., Bolten, A., Jung, A., Michels, R., and Soukkamäki, J. (2015). Low-weight and uav-based hyperspectral full-frame cameras for monitoring crops: Spectral comparison with portable spectroradiometer measurements. *Photogrammetrie-Fernerkundung-Geoinformation*, 2015(1):69–79.

[22] Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. (2008). Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359.

[23] Bellman, R. (2015). *Adaptive Control Processes: A Guided Tour*. Princeton Legacy Library. Princeton University Press.

[24] Benediktsson, J. A. and Ghamisi, P. (2015). *Spectral-spatial classification of hyperspectral remote sensing images*. Artech House.

[25] Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828.

[26] Bengio, Y., LeCun, Y., et al. (2007). Scaling learning algorithms towards ai. *Large-scale kernel machines*, 34(5):1–41.

[27] Benítez, J. M., Castro, J. L., and Requena, I. (1997). Are artificial neural networks black boxes? *IEEE Transactions on Neural Networks*, 8(5):1156–1164.

[28] Berger, M., Moreno, J., Muller, A., Schaepman, M., Wursteisen, P., Rast, M., and Attema, E. (2000). The digital airborne imaging spectrometer experiment-daisex'99. In *IGARSS 2000. IEEE 2000 International Geoscience and Remote Sensing Symposium. Taking the Pulse of the Planet: The Role of Remote Sensing in Managing the Environment. Proceedings (Cat. No. 00CH37120)*, volume 7, pages 3039–3041. IEEE.

[29] Bezdek, J. C. (1992). On the relationship between neural networks, pattern recognition and intelligence. *International journal of approximate reasoning*, 6(2):85–107.

[30] Bhardwaj, K. and Patra, S. (2018). An unsupervised technique for optimal feature selection in attribute profiles for spectral-spatial classification of hyperspectral images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 138:139 – 150.

[31] Billings, S. A. and Zheng, G. L. (1995). Radial basis function network configuration using genetic algorithms. *Neural Networks*, 8(6):877–890.

[32] Bioucas-Dias, J. M., Plaza, A., Camps-Valls, G., Scheunders, P., Nasrabadi, N., and Chanussot, J. (2013). Hyperspectral remote sensing data analysis and future challenges. *IEEE Geosci. Remote Sens. Mag.*, 1(2):6–36.

[33] Bioucas-Dias, J. M., Plaza, A., Dobigeon, N., Parente, M., Du, Q., Gader, P., and Chanussot, J. (2012). Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 5(2):354–379.

[34] Bishop, C. M. (2006). *Pattern recognition and machine learning*. springer.

[35] Blockeel, H. (2011). Hypothesis space. *Encyclopedia of Machine Learning*, 1:511–513.

[36] Blum, A. and Rivest, R. L. (1989). Training a 3-node neural network is np-complete. In *Advances in neural information processing systems*, pages 494–501.

[37] Boureau, Y.-L., Ponce, J., and LeCun, Y. (2010). A theoretical analysis of feature pooling in visual recognition. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 111–118.

[38] Braga, F., Giardino, C., Bassani, C., Matta, E., Candiani, G., Strömbeck, N., Adamo, M., and Bresciani, M. (2013). Assessing water quality in the northern adriatic sea from hico^{TM} data. *Remote sensing letters*, 4(10):1028–1037.

[39] Brando, V. E. and Dekker, A. G. (2003). Satellite hyperspectral remote sensing for estimating estuarine and coastal water quality. *IEEE transactions on geoscience and remote sensing*, 41(6):1378–1387.

[40] Briottet, X., Boucher, Y., Dimmeler, A., Malaplate, A., Cini, A., Diani, M., Bekman, H., Schwering, P., Skauli, T., Kasen, I., et al. (2006). Military applications of hyperspectral imagery. In *Targets and backgrounds XII: Characterization and representation*, volume 6239, page 62390B. International Society for Optics and Photonics.

[41] Bruce, L. M., Koger, C. H., and Li, J. (2002). Dimensionality reduction of hyperspectral data using discrete wavelet transform feature extraction. *IEEE Transactions on geoscience and remote sensing*, 40(10):2331–2338.

[42] Bue, B. D., Thompson, D. R., Eastwood, M., Green, R. O., Gao, B. C., Keymeulen, D., Sarture, C. M., Mazer, A. S., and Luong, H. H. (2015). Real-time atmospheric correction of aviris-ng imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 53(12):6419–6428.

[43] Calin, M. A., Parasca, S. V., and Manea, D. (2018). Comparison of spectral angle mapper and support vector machine classification methods for mapping skin burn using hyperspectral imaging. In *Unconventional Optical Imaging*, volume 10677, page 106773P. International Society for Optics and Photonics.

[44] Campbell, J. B. and Wynne, R. H. (2011). *Introduction to remote sensing*. Guilford Press.

[45] Camps-Valls, G. (2016). Kernel spectral angle mapper. *Electronics Letters*, 52(14):1218–1220.

[46] Camps-Valls, G., Tuia, D., Bruzzone, L., and Benediktsson, J. A. (2014). Advances in hyperspectral image classification: Earth monitoring with statistical learning methods. *IEEE signal processing magazine*, 31(1):45–54.

[47] Cariou, C. and Chehdi, K. (2015). Unsupervised nearest neighbors clustering with application to hyperspectral images. *IEEE Journal of Selected Topics in Signal Processing*, 9(6):1105–1116.

[48] Casey, R. G. and Nagy, G. (1971). Advances in pattern recognition. *Scientific American*, 224(4):56–71.

[49] Cavalli, R. M., Colosi, F., Palombo, A., Pignatti, S., and Poscolieri, M. (2007). Remote hyperspectral imagery as a support to archaeological prospection. *Journal of Cultural Heritage*, 8(3):272–283.

[50] Ceamanos, X. and Valero, S. (2016). Processing hyperspectral images. In *Optical Remote Sensing of Land Surface*, pages 163–200. Elsevier.

[51] Chabrillat, S., Milewski, R., Schmid, T., Rodriguez, M., Escribano, P., Pelayo, M., and Palacios-Orueta, A. (2014). Potential of hyperspectral imagery for the spatial assessment of soil erosion stages in agricultural semi-arid spain at different scales. In *2014 IEEE Geoscience and Remote Sensing Symposium*, pages 2918–2921.

[52] Champagne, C. M., Staenz, K., Bannari, A., McNairn, H., and Deguise, J.-C. (2003). Validation of a hyperspectral curve-fitting model for the estimation of plant water content of agricultural canopies. *Remote Sensing of Environment*, 87(2-3):148–160.

[53] Chang, C.-I. (2003). *Hyperspectral imaging: techniques for spectral detection and classification*, volume 1. Springer Science & Business Media.

[54] Chang, C.-I. (2006). Recent advances in hyperspectral signal and image processing. *Transworld Research Network*, pages 146–169.

[55] Chang, C.-I. (2007). *Hyperspectral data exploitation: theory and applications*. John Wiley & Sons.

[56] Charles, A. S., Olshausen, B. A., and Rozell, C. J. (2011). Learning sparse codes for hyperspectral imagery. *IEEE Journal of Selected Topics in Signal Processing*, 5(5):963–978.

[57] Chen, C., Jiang, J., Zhang, B., Yang, W., and Guo, J. (2015a). Hyperspectral image classification using gradient local auto-correlations. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 454–458. IEEE.

[58] Chen, C. H. and Ho, P.-G. P. (2008). Statistical pattern recognition in remote sensing. *Pattern Recognition*, 41(9):2731–2741.

[59] Chen, G. and Qian, S.-E. (2011). Denoising of hyperspectral imagery using principal component analysis and wavelet shrinkage. *IEEE Transactions on Geoscience and remote sensing*, 49(3):973–980.

[60] Chen, S. and Wang, Y. (2014). Convolutional neural network and convex optimization. *Dept. of Elect. and Comput. Eng., Univ. of California at San Diego, San Diego, CA, USA, Tech. Rep*.

[61] Chen, Y., Jiang, H., Li, C., Jia, X., and Ghamisi, P. (2016). Deep feature extraction and classification of hyperspectral images based on convolutional neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 54(10):6232–6251.

[62] Chen, Y., Lin, Z., Zhao, X., Wang, G., and Gu, Y. (2014). Deep learning-based classification of hyperspectral data. *IEEE Journal of Selected topics in applied earth observations and remote sensing*, 7(6):2094–2107.

[63] Chen, Y., Zhao, X., and Jia, X. (2015b). Spectral–spatial classification of hyperspectral data based on deep belief network. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8(6):2381–2392.

[64] Chen, Y., Zhu, L., Ghamisi, P., Jia, X., Li, G., and Tang, L. (2017). Hyperspectral images classification with gabor filtering and convolutional neural network. *IEEE Geoscience and Remote Sensing Letters*, 14(12):2355–2359.

[65] Cheng, G., Han, J., and Lu, X. (2017a). Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883.

[66] Cheng, T. and Zhan, X. (2017). Pattern recognition for predictive, preventive, and personalized medicine in cancer. *EPMA Journal*, 8(1):51–60.

[67] Cheng, Y., Wang, D., Zhou, P., and Zhang, T. (2017b). A survey of model compression and acceleration for deep neural networks. *arXiv preprint arXiv:1710.09282*.

[68] Chou, W. and Juang, B.-H. (2003). *Pattern recognition in speech and language processing*. Crc Press.

[69] Chow, C.-K. (1957). An optimum character recognition system using decision functions. *IRE Transactions on Electronic Computers*, EC-6(4):247–254.

[70] Christophe, E. (2011). Hyperspectral data compression tradeoff. In *Optical remote sensing*, pages 9–29. Springer.

[71] Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

[72] Chutia, D., Bhattacharyya, D., Sarma, K. K., Kalita, R., and Sudhakar, S. (2016). Hyperspectral remote sensing classifications: a perspective survey. *Transactions in GIS*, 20(4):463–490.

[73] Cocks, T., Jenssen, R., Stewart, A., Wilson, I., and Shields, T. (1998). The hymaptm airborne hyperspectral sensor: the system, calibration and performance. In *Proceedings of the 1st EARSeL workshop on Imaging Spectroscopy*, pages 37–42. EARSeL.

[74] Collobert, R. and Bengio, S. (2004). Links between perceptrons, mlps and svms. In *Proceedings of the twenty-first international conference on Machine learning*, page 23. ACM.

[75] Coops, N. C., Smith, M. L., Martin, M. E., and Ollinger, S. V. (2003). Prediction of eucalypt foliage nitrogen content from satellite-derived hyperspectral data. *IEEE Transactions on Geoscience and Remote Sensing*, 41(6):1338–1346.

[76] Coulter, D., Hauff, P., and Kerby, W. (2007). Airborne hyperspectral remote sensing. In *Proceedings of the 5th Decennial International Conference on Mineral Exploration*, pages 375–378. Toronto, Ontario, Canada.

[77] Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27.

[78] Crowley, J. K., Brickey, D. W., and Rowan, L. C. (1989). Airborne imaging spectrometer data of the ruby mountains, montana: mineral discrimination using relative absorption band-depth images. *Remote Sensing of Environment*, 29(2):121–134.

[79] Cutter, M. A., Johns, L. S., Lobb, D. R., Williams, T. L., and Settle, J. (2004). Flight experience of the compact high-resolution imaging spectrometer (chris). In *Imaging Spectrometry IX*, volume 5159, pages 392–405. International Society for Optics and Photonics.

[80] Cutter, M. A., Lobb, D. R., and Cockshott, R. A. (2000). Compact high resolution imaging spectrometer (chris). *Acta Astronautica*, 46(2-6):263–268.

[81] Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. IEEE.

[82] Dalla Mura, M., Atli Benediktsson, J., Waske, B., and Bruzzone, L. (2010). Extended profiles with morphological attribute filters for the analysis of hyperspectral data. *International Journal of Remote Sensing*, 31(22):5975–5991.

[83] Debes, C., Merentitis, A., Heremans, R., Hahn, J., Frangiadakis, N., van Kasteren, T., Liao, W., Bellens, R., Pižurica, A., Gautama, S., Philips, W., Prasad, S., Du, Q., and Pacifici, F. (2014). Hyperspectral and lidar data fusion: Outcome of the 2013 grss data fusion contest. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 7(6):2405–2418.

[84] Deisenroth, M. P., Faisal, A. A., and Ong, C. S. (2020). *Mathematics for machine learning*. Cambridge University Press.

[85] Derde, M.-P. and Massart, D. (1986). Supervised pattern recognition: the ideal method? *Analytica Chimica Acta*, 191:1–16.

[86] Dong, H., Li, T., Leng, J., Kong, L., and Bai, G. (2017). Gcn: Gpu-based cube cnn framework for hyperspectral image classification. In *2017 46th International Conference on Parallel Processing (ICPP)*, pages 41–49.

[87] Du, J. and Li, Z. (2018). A hyperspectral target detection framework with subtraction pixel pair features. *IEEE Access*, 6:45562–45577.

[88] Du, Q. and Chang, C.-I. (2001). A linear constrained distance-based discriminant analysis for hyperspectral image classification. *Pattern Recognition*, 34(2):361–373.

[89] Du, Q. and Yang, H. (2008). Similarity-based unsupervised band selection for hyperspectral image analysis. *IEEE geoscience and remote sensing letters*, 5(4):564–568.

[90] Duda, R. O., Hart, P. E., and Stork, D. G. (2012). *Pattern classification*. John Wiley & Sons.

[91] Dugas, C., Bengio, Y., Bélisle, F., Nadeau, C., and Garcia, R. (2001). Incorporating second-order functional knowledge for better option pricing. In *Advances in neural information processing systems*, pages 472–478.

[92] Eckardt, A., Horack, J., Lehmann, F., Krutz, D., Drescher, J., Whorton, M., and Soutullo, M. (2015). Desis (dlr earth sensing imaging spectrometer for the iss-muses platform). In *2015 IEEE international geoscience and remote sensing symposium (IGARSS)*, pages 1457–1459. IEEE.

[93] Eismann, M. T. and Hardie, R. C. (2005). Hyperspectral resolution enhancement using high-resolution multispectral imagery with arbitrary response functions. *IEEE Transactions on Geoscience and Remote Sensing*, 43(3):455–465.

[94] El-Magd, I. A. and El-Zeiny, A. (2014). Quantitative hyperspectral analysis for characterization of the coastal water from damietta to port said, egypt. *The Egyptian Journal of Remote Sensing and Space Science*, 17(1):61 – 76.

[95] Emery, W. and Camps, A. (2017). Basic electromagnetic concepts and applications to optical sensors. *Introduction to Satellite Remote Sensing; Elsevier: Amsterdam, The Netherlands*.

[96] Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., Vincent, P., and Bengio, S. (2010). Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb):625–660.

[97] Fang, B., Li, Y., Zhang, H., and Chan, J. (2018). Semi-supervised deep learning classification for hyperspectral image based on dual-strategy sample selection. *Remote Sensing*, 10(4):574.

[98] Fauvel, M., Benediktsson, J. A., Chanussot, J., and Sveinsson, J. R. (2008). Spectral and spatial classification of hyperspectral data using svms and morphological profiles. *IEEE Transactions on Geoscience and Remote Sensing*, 46(11):3804–3814.

[99] Fauvel, M., Tarabalka, Y., Benediktsson, J. A., Chanussot, J., and Tilton, J. C. (2013). Advances in spectral-spatial classification of hyperspectral images. *Proceedings of the IEEE*, 101(3):652–675.

[100] Feingersh, T. and Dor, E. B. (2015). Shalom–a commercial hyperspectral space mission. *Optical payloads for space missions*, pages 247–263.

[101] Field, D. J. (1999). Wavelets, vision and the statistics of natural scenes. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 357(1760):2527–2542.

[102] Fischer, W. A., Hemphill, W., and Kover, A. (1976). Progress in remote sensing (1972–1976). *Photogrammetria*, 32(2):33–72.

[103] Fisher, P. (1997). The pixel: a snare and a delusion. *International Journal of Remote Sensing*, 18(3):679–685.

[104] Fogler, H. R. (1974). A pattern recognition model for forecasting. *Management science*, 20(8):1178–1189.

[105] Fowler, J. E. (2014). Compressive pushbroom and whiskbroom sensing for hyperspectral remote-sensing imaging. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 684–688. IEEE.

[106] Fu, K. (1968). *Sequential methods in pattern recognition and machine learning*. Academic press.

[107] Fu, K. S. (1979). *Application of pattern recognition to remote sensing*. School of Electrical Engineering, Purdue University.

[108] Fukunaga, K. (2013). *Introduction to Statistical Pattern Recognition*. Computer science and scientific computing. Elsevier Science, Boston.

[109] Fukunaga, K. and Hummels, D. M. (1989). Leave-one-out procedures for nonparametric error estimates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(4):421–423.

[110] Funahashi, K.-i. (1998). Multilayer neural networks and bayes decision theory. *Neural Networks*, 11(2):209–213.

[111] Fussell, J., Rundquist, D., and Harrington, J. (1986). On defining remote sensing. *Photogrammetric Engineering and Remote Sensing*, 52(9):1507–1511.

[112] Gao, B.-C., Montes, M. J., Davis, C. O., and Goetz, A. F. (2009). Atmospheric correction algorithms for hyperspectral remote sensing data of land and ocean. *Remote Sensing of Environment*, 113:S17–S24.

[113] Geva, S. and Sitte, J. (1991). Adaptive nearest neighbor pattern classification. *IEEE Trans. on Neural Networks*, 2(0):2.

[114] Gewali, U. B., Monteiro, S. T., and Saber, E. (2018). Machine learning based hyperspectral image analysis: a survey. *arXiv preprint arXiv:1802.08701*.

[115] Ghamisi, P., Maggiori, E., Li, S., Souza, R., Tarablaka, Y., Moser, G., De Giorgi, A., Fang, L., Chen, Y., Chi, M., et al. (2018). New frontiers in spectral-spatial hyperspectral image classification: The latest advances based on mathematical morphology, markov random fields, segmentation, sparse representation, and deep learning. *IEEE Geoscience and Remote Sensing Magazine*, 6(3):10–43.

[116] Ghamisi, P., Plaza, J., Chen, Y., Li, J., and Plaza, A. J. (2017a). Advanced spectral classifiers for hyperspectral images: A review. *IEEE Geoscience and Remote Sensing Magazine*, 5(1):8–32.

[117] Ghamisi, P., Yokoya, N., Li, J., Liao, W., Liu, S., Plaza, J., Rasti, B., and Plaza, A. (2017b). Advances in hyperspectral image and signal processing: A comprehensive overview of the state of the art. *IEEE Geoscience and Remote Sensing Magazine*, 5(4):37–78.

[118] Goel, P., Prasher, S., Patel, R., Landry, J., Bonnell, R., and Viau, A. (2003). Classification of hyperspectral data by decision trees and artificial neural networks to identify weed stress and nitrogen status of corn. *Computers and Electronics in Agriculture*, 39(2):67 – 93.

[119] Goetz, A. F. (2009). Three decades of hyperspectral remote sensing of the earth: A personal view. *Remote Sensing of Environment*, 113:S5–S16.

[120] Goetz, A. F., Vane, G., Solomon, J. E., and Rock, B. N. (1985). Imaging spectrometry for earth remote sensing. *science*, 228(4704):1147–1153.

[121] Gomez, C., Drost, A., and Roger, J.-M. (2015). Analysis of the uncertainties affecting predictions of clay contents from vnir/swir hyperspectral data. *Remote Sensing of Environment*, 156:58 – 70.

[122] Gomez, R. B. and Dasgupta, S. (2004). Use of hyperspectral remote sensing for detection and monitoring of chemical and biological agents: a survey. In *Chemical and Biological Standoff Detection II*, volume 5584, pages 276–285. International Society for Optics and Photonics.

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

[124] Green, R. O., Eastwood, M. L., Sarture, C. M., Chrien, T. G., Aronsson, M., Chippendale, B. J., Faust, J. A., Pavri, B. E., Chovit, C. J., Solis, M., et al. (1998). Imaging spectroscopy and the airborne visible/infrared imaging spectrometer (aviris). *Remote sensing of environment*, 65(3):227–248.

[125] Große-Stoltenberg, A., Hellmann, C., Werner, C., Oldeland, J., and Thiele, J. (2016). Evaluation of continuous vnir-swir spectra versus narrowband hyperspectral indices to discriminate the invasive acacia longifolia within a mediterranean dune ecosystem. *Remote Sensing*, 8(4).

[126] Guanter, L., Kaufmann, H., Segl, K., Foerster, S., Rogass, C., Chabrillat, S., Kuester, T., Hollstein, A., Rossner, G., Chlebek, C., et al. (2015). The enmap spaceborne imaging spectroscopy mission for earth observation. *Remote Sensing*, 7(7):8830–8857.

[127] Guo, B., Gunn, S. R., Damper, R. I., and Nelson, J. D. (2006). Band selection for hyperspectral image classification using mutual information. *IEEE Geoscience and Remote Sensing Letters*, 3(4):522–526.

[128] Haboudane, D., Miller, J. R., Pattey, E., Zarco-Tejada, P. J., and Strachan, I. B. (2004). Hyperspectral vegetation indices and novel algorithms for predicting green lai of crop canopies: Modeling and validation in the context of precision agriculture. *Remote sensing of environment*, 90(3):337–352.

[129] Hagen, N. A. and Kudenov, M. W. (2013). Review of snapshot spectral imaging technologies. *Optical Engineering*, 52(9):090901.

[130] Ham, J., Chen, Y., Crawford, M. M., and Ghosh, J. (2005). Investigation of the random forest framework for classification of hyperspectral data. *IEEE Transactions on Geoscience and Remote Sensing*, 43(3):492–501.

[131] Hassoun, M. H. et al. (1995). *Fundamentals of artificial neural networks*. MIT press.

[132] Haut, J., Paoletti, M., Plaza, J., and Plaza, A. (2019a). Hyperspectral image classification using random occlusion data augmentation. *IEEE Geoscience and Remote Sensing Letters*.

[133] Haut, J. M., Bernabé, S., Paoletti, M. E., Fernandez-Beltran, R., Plaza, A., and Plaza, J. (2019b). Low–high-power consumption architectures for deep-learning models applied to hyperspectral image classification. *IEEE Geoscience and Remote Sensing Letters*, 16(5):776–780.

[134] Haut, J. M., Paoletti, M., Paz-Gallardo, A., Plaza, J., and Plaza, A. (2017a). Cloud implementation of logistic regression for hyperspectral image classification. In *Proceedings of the 17th International Conference on Computational and Mathematical Methods in Science and Engineering, CMMSE*, pages 1063–2321.

[135] Haut, J. M., Paoletti, M., Plaza, J., and Plaza, A. (2017b). Cloud implementation of the k-means algorithm for hyperspectral image analysis. *The Journal of Supercomputing*, 73(1):514–529.

[136] Haut, J. M., Paoletti, M. E., Plaza, J., Li, J., and Plaza, A. (2018a). Active learning with convolutional neural networks for hyperspectral image classification using a new bayesian approach. *IEEE Transactions on Geoscience and Remote Sensing*, pages 1–22.

[137] Haut, J. M., Paoletti, M. E., Plaza, J., and Plaza, A. (2018b). Fast dimensionality reduction and classification of hyperspectral images with extreme learning machines. *Journal of Real-Time Image Processing*, pages 1–24.

[138] Haut, J. M., Paoletti, M. E., Plaza, J., Plaza, A., and Li, J. (2019). Visual attention-driven hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, pages 1–16.

[139] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034.

[140] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1904–1916.

[141] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

[142] He, L., Li, J., Plaza, A., and Li, Y. (2017a). Discriminative low-rank gabor filtering for spectral-spatial hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(3):1381–1395.

[143] He, N., Paoletti, M. E., n. M. Haut, J., Fang, L., Li, S., Plaza, A., and Plaza, J. (2018). Feature extraction with multiscale covariance maps for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, pages 1–15.

[144] He, Z., Liu, H., Wang, Y., and Hu, J. (2017b). Generative adversarial networks-based semi-supervised learning for hyperspectral image classification. *Remote Sensing*, 9(10):1042.

[145] Heiden, U., Heldens, W., Roessner, S., Segl, K., Esch, T., and Mueller, A. (2012). Urban structure type characterization using hyperspectral remote sensing and height information. *Landscape and urban Planning*, 105(4):361–375.

[146] Heldens, W., Heiden, U., Esch, T., Stein, E., and Müller, A. (2011). Can the future enmap mission contribute to urban applications? a literature survey. *Remote Sensing*, 3(9):1817–1846.

[147] Hendrix, E. M., García, I., Plaza, J., and Plaza, A. (2010). Minimum volume simplicial enclosure for spectral unmixing of remotely sensed hyperspectral data. In *2010 IEEE International Geoscience and Remote Sensing Symposium*, pages 193–196. IEEE.

[148] Heumann, B. W. (2011). Satellite remote sensing of mangrove forests: Recent advances and future opportunities. *Progress in Physical Geography*, 35(1):87–108.

[149] Heylen, R., Parente, M., and Gader, P. (2014). A review of nonlinear hyperspectral unmixing methods. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 7(6):1844–1868.

[150] Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Comput.*, 18(7):1527–1554.

[151] Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

[152] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

[153] Hong, D., Wu, X., Ghamisi, P., Chanussot, J., Yokoya, N., and Zhu, X. X. (2020). Invariant attribute profiles: A spatial-frequency joint feature extractor for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*.

[154] Hu, W., Huang, Y., Wei, L., Zhang, F., and Li, H. (2015). Deep convolutional neural networks for hyperspectral image classification. *Journal of Sensors*, 2015.

[155] Huadong, G., Jianmin, X., Guoqiang, N., and Jialing, M. (2001). A new airborne earth observing system and its applications. In *IGARSS 2001. Scanning the Present and Resolving the Future. Proceedings. IEEE 2001 International Geoscience and Remote Sensing Symposium (Cat. No.01CH37217)*, volume 1, pages 549–551 vol.1.

[156] Huang, R. and He, M. (2005). Band selection based on feature weighting for classification of hyperspectral data. *IEEE Geoscience and Remote Sensing Letters*, 2(2):156–159.

[157] Huang, X. and Zhang, L. (2013). An svm ensemble approach combining spectral, structural, and semantic features for the classification of high-resolution remotely sensed imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 51(1):257–272.

[158] Hughes, G. (1968). On the mean accuracy of statistical pattern recognizers. *IEEE Transactions on Information Theory*, 14(1):55–63.

[159] Hulley, G. C., Ghent, D., and Hook, S. J. (2019). A look to the future: Thermal-infrared missions and measurements. In *Taking the Temperature of the Earth*, pages 227–237. Elsevier.

[160] Isermann, R. (1997). Supervision, fault-detection and fault-diagnosis methods—an introduction. *Control engineering practice*, 5(5):639–652.

[161] Ishida, T., Kurihara, J., Viray, F. A., Namuco, S. B., Paringit, E. C., Perez, G. J., Takahashi, Y., and Marciano Jr, J. J. (2018). A novel approach for vegetation classification using uav-based hyperspectral imaging. *Computers and electronics in agriculture*, 144:80–85.

[162] Iwasaki, A., Ohgi, N., Tanii, J., Kawashima, T., and Inada, H. (2011). Hyperspectral imager suite (hisui)-japanese hyper-multi spectral radiometer. In *2011 IEEE International Geoscience and Remote Sensing Symposium*, pages 1025–1028. IEEE.

[163] Jain, A. K. (2010). Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666.

[164] Jiménez, L. O., Rivera-Medina, J. L., Rodríguez-Díaz, E., Arzuaga-Cruz, E., and Ramírez-Vélez, M. (2005). Integration of spatial and spectral information by means of unsupervised extraction and classification for homogenous objects applied to multispectral and hyperspectral data. *IEEE Transactions on Geoscience and Remote Sensing*, 43(4):844–851.

[165] Jing, X., Leigh, L., Helder, D., Pinto, C. T., and Aaron, D. (2019). Lifetime absolute calibration of the eo-1 hyperion sensor and its validation. *IEEE Transactions on Geoscience and Remote Sensing*, 57(11):9466–9475.

[166] Jolliffe, I. (2002). *Principal Component Analysis*. Springer Series in Statistics. Springer.

[167] Kailath, T. (1967). The divergence and bhattacharyya distance measures in signal selection. *IEEE transactions on communication technology*, 15(1):52–60.

[168] Kallepalli, A., Kumar, A., and Khoshelham, K. (2014). Entropy based determination of optimal principal components of airborne prism experiment (apex) imaging spectrometer data for improved land cover classification. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 40(8):781.

[169] Kang, X., Li, C., Li, S., and Lin, H. (2018). Classification of hyperspectral images by gabor filtering based deep network. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 11(4):1166–1178.

[170] Kang, X., Zhang, X., Li, S., Li, K., Li, J., and Benediktsson, J. A. (2017). Hyperspectral anomaly detection with attribute and edge-preserving filters. *IEEE Transactions on Geoscience and Remote Sensing*, 55(10):5600–5611.

[171] Kang, X., Zhuo, B., and Duan, P. (2018). Dual-path network-based hyperspectral image classification. *IEEE Geoscience and Remote Sensing Letters*, pages 1–5.

[172] Keshava, N. (2004). Distance metrics and band selection in hyperspectral processing with applications to material identification and spectral libraries. *IEEE Transactions on Geoscience and remote sensing*, 42(7):1552–1565.

[173] khalil Alsmadi, M., Omar, K. B., Noah, S. A., and Almarashdah, I. (2009). Performance comparison of multi-layer perceptron (back propagation, delta rule and perceptron) algorithms in neural networks. In *2009 IEEE International Advance Computing Conference*, pages 296–299. IEEE.

[174] Khan, M. J., Khan, H. S., Yousaf, A., Khurshid, K., and Abbas, A. (2018). Modern trends in hyperspectral image analysis: A review. *IEEE Access*, 6:14118–14129.

[175] Klambauer, G., Unterthiner, T., Mayr, A., and Hochreiter, S. (2017). Self-normalizing neural networks. In *Advances in Neural Information Processing Systems*, pages 971–980.

[176] Kobayashi, T. and Otsu, N. (2008). Image feature extraction using gradient local auto-correlations. In *European conference on computer vision*, pages 346–358. Springer.

[177] Koch, B. (2010). Status and future of laser scanning, synthetic aperture radar and hyperspectral remote sensing data for forest biomass assessment. *ISPRS Journal of Photogrammetry and Remote Sensing*, 65(6):581–590.

[178] Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480.

[179] Koponen, S., Pulliainen, J., Kallio, K., and Hallikainen, M. (2002). Lake water quality classification with airborne hyperspectral spectrometer and simulated meris data. *Remote Sensing of Environment*, 79(1):51 – 59.

[180] Kotsiantis, S. B., Zaharakis, I., and Pintelas, P. (2007). Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160:3–24.

[181] Kotsiantis, S. B., Zaharakis, I. D., and Pintelas, P. E. (2006). Machine learning: a review of classification and combining techniques. *Artificial Intelligence Review*, 26(3):159–190.

[182] Krishna, K. and Murty, M. N. (1999). Genetic k-means algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 29(3):433–439.

[183] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

[184] Kruse, F. A. (1988). Use of airborne imaging spectrometer data to map minerals associated with hydrothermally altered rocks in the northern grapevine mountains, nevada, and california. *Remote Sensing of Environment*, 24(1):31–51.

[185] Kuching, S. (2007). The performance of maximum likelihood, spectral angle mapper, neural network and decision tree classifiers in hyperspectral image analysis. *Journal of Computer Science*, 3(6):419–423.

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

[187] Lachenbruch, P. A. and Mickey, M. R. (1968). Estimation of error rates in discriminant analysis. *Technometrics*, 10(1):1–11.

[188] Landgrebe, D. (2002). Hyperspectral image data analysis. *IEEE Signal processing magazine*, 19(1):17–28.

[189] Landgrebe, D. A. (2005). *Signal theory methods in multispectral remote sensing*, volume 29. John Wiley & Sons.

[190] Larochelle, H. and Bengio, Y. (2008). Classification using Discriminative Restricted Boltzmann Machines. In *Proceedings of the 25th international conference on Machine learning - ICML '08*, page 536.

[191] Larsson, G., Maire, M., and Shakhnarovich, G. (2016). Fractalnet: Ultra-deep neural networks without residuals. *arXiv preprint arXiv:1605.07648*.

[192] LeCun, Y. (2019). Deep learning hardware: Past, present, and future. In *2019 IEEE International Solid-State Circuits Conference-(ISSCC)*, pages 12–19. IEEE.

[193] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436.

[194] Ledley, R. (1969). Automatic pattern recognition for clinical medicine. *Proceedings of the IEEE*, 57(11):2017–2035.

[195] Lee, C. A., Gasster, S. D., Plaza, A., Chang, C.-I., and Huang, B. (2011). Recent developments in high performance computing for remote sensing: A review. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 4(3):508–527.

[196] Lee, C. M., Cable, M. L., Hook, S. J., Green, R. O., Ustin, S. L., Mandl, D. J., and Middleton, E. M. (2015). An introduction to the nasa hyperspectral infrared imager (hyspiri) mission and preparatory activities. *Remote Sensing of Environment*, 167:6–19.

[197] Lei, T., Zhang, Y., Wang, S. I., Dai, H., and Artzi, Y. (2017). Simple recurrent units for highly parallelizable recurrence. *arXiv preprint arXiv:1709.02755*.

[198] Li, J. (2015). Active learning for hyperspectral image classification with a stacked autoencoders based neural network. In *2015 7th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, pages 1–4.

[199] Li, J., Bioucas-Dias, J. M., and Plaza, A. (2010). Semisupervised hyperspectral image segmentation using multinomial logistic regression with active learning. *IEEE Transactions on Geoscience and Remote Sensing*, 48(11):4085–4098.

[200] Li, J., Bioucas-Dias, J. M., and Plaza, A. (2011). Hyperspectral image segmentation using a new bayesian approach with active learning. *IEEE Transactions on Geoscience and Remote Sensing*, 49(10):3947–3960.

[201] Li, J., Bioucas-Dias, J. M., and Plaza, A. (2012). Spectral–spatial hyperspectral image segmentation using subspace multinomial logistic regression and markov random fields. *IEEE Transactions on Geoscience and Remote Sensing*, 50(3):809–823.

[202] Li, S., Song, W., Fang, L., Chen, Y., Ghamisi, P., and Benediktsson, J. A. (2019). Deep learning for hyperspectral image classification: An overview. *IEEE Transactions on Geoscience and Remote Sensing*.

[203] Li, W., Chen, C., Su, H., and Du, Q. (2015). Local binary patterns and extreme learning machine for hyperspectral imagery classification. *IEEE Transactions on Geoscience and Remote Sensing*, 53(7):3681–3693.

[204] Li, W. and Du, Q. (2016). A survey on representation-based classification and detection in hyperspectral remote sensing imagery. *Pattern Recognition Letters*, 83:115–123.

[205] Liang, H. and Li, Q. (2016). Hyperspectral imagery classification using sparse representations of convolutional neural network features. *Remote Sensing*, 8(2):99.

[206] Lichtman, A. J. and Keilis-Borok, V. I. (1981). Pattern recognition applied to presidential elections in the united states, 1860-1980: Role of integral social, economic, and political traits. *Proceedings of the National Academy of Sciences of the United States of America*, 78(11):7230.

[207] Likas, A., Vlassis, N., and Verbeek, J. J. (2003). The global k-means clustering algorithm. *Pattern recognition*, 36(2):451–461.

[208] Lipton, Z. C. (2016). The mythos of model interpretability. *arXiv preprint arXiv:1606.03490*.

[209] Lissack, T. and Fu, K.-S. (1976). Error estimation in pattern recognition via l_alpha-distance between posterior density functions. *IEEE Transactions on Information Theory*, 22(1):34–45.

[210] Liu, C. and Wechsler, H. (1998). Enhanced fisher linear discriminant models for face recognition. In *Proceedings. Fourteenth International Conference on Pattern Recognition (Cat. No. 98EX170)*, volume 2, pages 1368–1372. IEEE.

[211] Liu, P., Zhang, H., and Eom, K. B. (2016). Active deep learning for classification of hyperspectral images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 10(2):712–724.

[212] Lucas, R., Rowlands, A., Niemann, O., and Merton, R. (2004). *Advanced Image Processing Techniques for Remotely Sensed Hyperspectral Data*. Springer Berlin Heidelberg, Berlin, Heidelberg.

[213] Lulla, V. (2009). Hyperspectral applications in urban geography. In Gatrell, J. D. and Jensen, R. R., editors, *Planning and Socioeconomic Applications*, pages 79–86. Springer Netherlands, Dordrecht.

[214] Ma, X., Wang, H., and Geng, J. (2016). Spectral–spatial classification of hyperspectral image based on deep auto-encoder. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(9):4073–4085.

[215] Maas, A. L., Hannun, A. Y., and Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3.

[216] Maffei, A., Haut, J. M., Paoletti, M. E., Plaza, J., Bruzone, L., and Plaza, A. (2019a). Efficient convolutional neural network for spectral-spatial hyperspectral denoising. In *2019 10th Workshop on Hyperspectral Imaging and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, pages 1–4. IEEE.

[217] Maffei, A., Haut, J. M., Paoletti, M. E., Plaza, J., Bruzzone, L., and Plaza, A. (2019b). A single model cnn for hyperspectral image denoising. *IEEE Transactions on Geoscience and Remote Sensing*.

[218] Makantasis, K., Karantzalos, K., Doulamis, A., and Doulamis, N. (2015). Deep supervised learning for hyperspectral data classification through convolutional neural networks. In *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 4959–4962. IEEE.

[219] Makhoul, J. (1991). Pattern recognition properties of neural networks. In *Neural Networks for Signal Processing Proceedings of the 1991 IEEE Workshop*, pages 173–187. IEEE.

[220] Manolakis, D., Marden, D., Shaw, G. A., et al. (2003). Hyperspectral image processing for automatic target detection applications. *Lincoln laboratory journal*, 14(1):79–116.

[221] Manovich, L. (2011). From reading to pattern recognition. *Lev Manovich*.

[222] Markham, B. L., Dabney, P. W., Reuter, D., Thome, K. J., Irons, J. R., Barsi, J. A., and Montanaro, M. (2011). Landsat data continuity mission operational land imager and thermal infrared sensor performance. In *Earth Observing Systems XVI*, volume 8153, page 81530D. International Society for Optics and Photonics.

[223] Martin, J. J. (1967). *Bayesian decision problems and Markov chains*. Wiley.

[224] Maxwell, A. E., Warner, T. A., and Fang, F. (2018). Implementation of machine-learning classification in remote sensing: An applied review. *International Journal of Remote Sensing*, 39(9):2784–2817.

[225] Mei, S., Ji, J., Bi, Q., Hou, J., Du, Q., and Li, W. (2016). Integrating spectral and spatial information into deep convolutional neural networks for hyperspectral classification. In *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 5067–5070.

[226] Melgani, F. and Bruzzone, L. (2004). Classification of hyperspectral remote sensing images with support vector machines. *IEEE Transactions on geoscience and remote sensing*, 42(8):1778–1790.

[227] Monteiro, S. T., Minekawa, Y., Kosugi, Y., Akazawa, T., and Oda, K. (2007). Prediction of sweetness and amino acid content in soybean crops from hyperspectral imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 62(1):2–12.

[228] Mookambiga, A. and Gomathi, V. (2016). Comprehensive review on fusion techniques for spatial information enhancement in hyperspectral imagery. *Multidimensional Systems and Signal Processing*, 27(4):863–889.

[229] Mou, L., Ghamisi, P., and Zhu, X. X. (2017). Deep recurrent neural networks for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(7):3639–3655.

[230] Mouroulis, P., Green, R. O., and Chrien, T. G. (2000). Design of pushbroom imaging spectrometers for optimum recovery of spectroscopic and spatial information. *Applied Optics*, 39(13):2210–2220.

[231] Mouroulis, P., Van Gorp, B., Green, R. O., Dierssen, H., Wilson, D. W., Eastwood, M., Boardman, J., Gao, B.-C., Cohen, D., Franklin, B., et al. (2014). Portable remote imaging spectrometer coastal ocean sensor: design, characteristics, and first flight results. *Applied optics*, 53(7):1363–1380.

[232] Mulla, D. J. (2013). Twenty five years of remote sensing in precision agriculture: Key advances and remaining knowledge gaps. *Biosystems engineering*, 114(4):358–371.

[233] Mura, M. D., Benediktsson, J. A., Waske, B., and Bruzzone, L. (2010). Morphological attribute profiles for the analysis of very high resolution images. *IEEE Transactions on Geoscience and Remote Sensing*, 48(10):3747–3762.

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

[235] Narendra, G. and Sivakumar, D. (2019). Deep learning based hyperspectral image analysis–a survey. *Journal of Computational and Theoretical Nanoscience*, 16(4):1528–1535.

[236] Nascimento, J. M. and Dias, J. M. (2005). Vertex component analysis: A fast algorithm to unmix hyperspectral data. *IEEE transactions on Geoscience and Remote Sensing*, 43(4):898–910.

[237] Nguyen, Q. and Hein, M. (2018). Optimization landscape and expressivity of deep cnns. In *International Conference on Machine Learning*, pages 3727–3736.

[238] Nogueira, K., Penatti, O. A., and dos Santos, J. A. (2017). Towards better exploiting convolutional neural networks for remote sensing scene classification. *Pattern Recognition*, 61:539–556.

[239] Numata, I., Roberts, D. A., Chadwick, O. A., Schimel, J. P., Galvão, L. S., and Soares, J. V. (2008). Evaluation of hyperspectral data for pasture estimate in the brazilian amazon using field and imaging spectrometers. *Remote Sensing of Environment*, 112(4):1569–1583.

[240] Olmanson, L. G., Brezonik, P. L., and Bauer, M. E. (2013). Airborne hyperspectral remote sensing to assess spatial distribution of water quality characteristics in large rivers: The mississippi river and its tributaries in minnesota. *Remote Sensing of Environment*, 130:254 – 265.

[241] O'Shea, K. and Nash, R. (2015). An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*.

[242] Paoletti, M., Haut, J., Plaza, J., and Plaza, A. (2018a). Deep&dense convolutional neural network for hyperspectral image classification. *Remote Sensing*, 10(9):1454.

[243] Paoletti, M., Haut, J., Plaza, J., and Plaza, A. (2019). Deep learning classifiers for hyperspectral imaging: A review. *ISPRS Journal of Photogrammetry and Remote Sensing*, 158:279–317.

[244] Paoletti, M. E., Haut, J. M., Fernandez-Beltran, R., Plaza, J., Plaza, A. J., and Pla, F. (2018b). Capsule networks for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, pages 1–16.

[245] Paoletti, M. E., Haut, J. M., Fernandez-Beltran, R., Plaza, J., Plaza, A. J., and Pla, F. (2019). Deep pyramidal residual networks for spectral–spatial hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 57(2):740–754.

[246] Paoletti, M. E., Haut, J. M., Plaza, J., and Plaza, A. (2018c). A new deep convolutional neural network for fast hyperspectral image classification. *ISPRS Journal of Photogrammetry and Remote Sensing*, 145:120 – 147. Deep Learning RS Data.

[247] Paoletti, M. E., Haut, J. M., Plaza, J., and Plaza, A. (2019a). Neighboring region dropout for hyperspectral image classification. *IEEE Geoscience and Remote Sensing Letters*.

[248] Paoletti, M. E., Haut, J. M., Plaza, J., and Plaza, A. (2019b). Neural ordinary differential equations for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*.

[249] Paoletti, M. E., Haut, J. M., Plaza, J., and Plaza, A. (2020a). Scalable recurrent neural network for hyperspectral image classification. *The Journal of Supercomputing*, pages 1–17.

[250] Paoletti, M. E., Haut, J. M., Tao, X., Miguel, J. P., and Plaza, A. (2020b). A new gpu implementation of support vector machines for fast hyperspectral image classification. *Remote Sensing*, 12(8):1257.

[251] Paoletti, M. E., m. Haut, J., Plaza, J., and Plaza, A. (2018). An investigation on self-normalized deep neural networks for hyperspectral image classification. In *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, pages 3607–3610.

[252] Parzen, E. (1962). On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076.

[253] Pasquinelli, M. (2019). How a machine learns and fails. *spheres: Journal for Digital Cultures*, -(5):1–17.

[254] Patrick, E. and Hancock, J. (1966). Nonsupervised sequential classification and recognition of patterns. *IEEE Transactions on Information Theory*, 12(3):362–372.

[255] Pearlman, J. S., Barry, P. S., Segal, C. C., Shepanski, J., Beiso, D., and Carman, S. L. (2003). Hyperion, a space-based imaging spectrometer. *IEEE Transactions on Geoscience and Remote Sensing*, 41(6):1160–1173.

[256] Pedamonti, D. (2018). Comparison of non-linear activation functions for deep neural networks on mnist classification task. *arXiv preprint arXiv:1804.02763*.

[257] Peerbhay, K. Y., Mutanga, O., and Ismail, R. (2015). Random forests unsupervised classification: The detection and mapping of solanum mauritianum infestations in plantation forestry using hyperspectral data. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8(6):3107–3122.

[258] Pengra, B. W., Johnston, C. A., and Loveland, T. R. (2007). Mapping an invasive plant, *Phragmites australis*, in coastal wetlands using the eo-1 hyperion hyperspectral sensor. *Remote Sensing of Environment*, 108(1):74–81.

[259] Penna, B., Tillo, T., Magli, E., and Olmo, G. (2007). Transform coding techniques for lossy hyperspectral data compression. *IEEE Transactions on Geoscience and Remote Sensing*, 45(5):1408–1421.

[260] Petersson, H., Gustafsson, D., and Bergstrom, D. (2016). Hyperspectral image analysis using deep learning - a review. In *2016 Sixth International Conference on Image Processing Theory, Tools and Applications (IPTA)*, pages 1–6. IEEE.

[261] Pignatti, S., Palombo, A., Pascucci, S., Romano, F., Santini, F., Simoniello, T., Umberto, A., Vincenzo, C., Acito, N., Diani, M., et al. (2013). The prisma hyperspectral mission: Science activities and opportunities for agriculture and land monitoring. In *2013 IEEE International Geoscience and Remote Sensing Symposium-IGARSS*, pages 4558–4561. IEEE.

[262] Plaza, A., Benediktsson, J. A., Boardman, J. W., Brazile, J., Bruzzone, L., Camps-Valls, G., Chanussot, J., Fauvel, M., Gamba, P., Gualtieri, A., et al. (2009). Recent advances in techniques for hyperspectral image processing. *Remote sensing of environment*, 113:S110–S122.

[263] Plaza, A., Martínez, P., Plaza, J., and Pérez, R. (2005). Dimensionality reduction and classification of hyperspectral image data using sequences of extended morphological transformations. *IEEE Transactions on Geoscience and remote sensing*, 43(3):466–479.

[264] Plaza, J., Plaza, A., Pérez, R., and Martínez, P. (2008). Parallel classification of hyperspectral images using neural networks. In *Computational Intelligence for Remote Sensing*, pages 193–216. Springer.

[265] Prieto, A., Atencia, M., and Sandoval, F. (2013). Advances in artificial neural networks and machine learning.

[266] Qian, S.-E. (2004). Hyperspectral data compression using a fast vector quantization algorithm. *IEEE Transactions on Geoscience and Remote Sensing*, 42(8):1791–1798.

[267] Qian, S.-E. (2015). *Optical payloads for space missions*. John Wiley & Sons.

[268] Qian, S.-E. (2020). *Hyperspectral Satellites and System Design*. CRC Press.

[269] Ramachandran, P., Zoph, B., and Le, Q. V. (2017). Swish: a self-gated activation function. *arXiv preprint arXiv:1710.05941*, 7.

[270] Randhe, P. H., Durbha, S. S., and Younan, N. H. (2016). Embedded high performance computing for on-board hyperspectral image classification. In *2016 8th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, pages 1–5.

[271] Rast, M. and Painter, T. H. (2019). Earth observation imaging spectroscopy for terrestrial systems: An overview of its history, techniques, and applications of its missions. *Surveys in Geophysics*, 40(3):303–331.

[272] Rasti, B., Hong, D., Hang, R., Ghamisi, P., Kang, X., Chanussot, J., and Benediktsson, J. A. (2020). Feature extraction for hyperspectral imagery: The evolution from shallow to deep. *arXiv preprint arXiv:2003.02822*.

[273] Rasti, B., Scheunders, P., Ghamisi, P., Licciardi, G., and Chanussot, J. (2018). Noise reduction in hyperspectral imagery: Overview and application. *Remote Sensing*, 10(3):482.

[274] Reichmuth, A. (2013). *Detection of forest parameters using imaging spectroscopy*. PhD thesis, University of Applied Sciences (HTW) Dresden.

[275] Resmini, R. G., Kappus, M. E., Aldrich, W. S., Harsanyi, J. C., and Anderson, M. (1997). Mineral mapping with hyperspectral digital imagery collection experiment (hydice) sensor data at cuprite, nevada, u.s.a. *International Journal of Remote Sensing*, 18(7):1553–1570.

[276] Richards, J. A. (1993). Sources and characteristics of remote sensing image data. In *Remote Sensing Digital Image Analysis*, pages 1–37. Springer.

[277] Richter, R. (2005). Hyperspectral sensors for military applications. Technical report, German Aerospace Center Wessling (DLR), Wessling (Germany).

[278] Rickard, L. J., Basedow, R. W., Zalewski, E. F., Silverglate, P. R., and Landers, M. (1993). Hydice: An airborne system for hyperspectral imaging. In *Imaging Spectrometry of the Terrestrial Environment*, volume 1937, pages 173–180. International Society for Optics and Photonics.

[279] Rigollet, P. (2015). Mathematics of machine learning. massachusetts institute of technology, cambridge, ma, usa.

[280] Rippel, O., Snoek, J., and Adams, R. P. (2015). Spectral representations for convolutional neural networks. In *Advances in neural information processing systems*, pages 2449–2457.

[281] Roberts, D. A., Dennison, P. E., Gardner, M. E., Hetzel, Y., Ustin, S. L., and Lee, C. T. (2003). Evaluation of the potential of hyperion for fire danger assessment by comparison to the airborne visible/infrared imaging spectrometer. *IEEE Transactions on Geoscience and Remote Sensing*, 41(6):1297–1310.

[282] Rodarmel, C. and Shan, J. (2002). Principal component analysis for hyperspectral image classification. *Surveying and Land Information Science*, 62(2):115–122.

[283] Rodríguez-Pérez, J. R., Riaño, D., Carlisle, E., Ustin, S., and Smart, D. R. (2007). Evaluation of hyperspectral reflectance indexes to detect grapevine water status in vineyards. *American Journal of Enology and Viticulture*, 58(3):302–317.

[284] Rojas, R. (1996). The backpropagation algorithm. In *Neural networks*, pages 149–182. Springer.

[285] Romero, A., Gatta, C., and Camps-Valls, G. (2016). Unsupervised deep feature extraction for remote sensing image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 54(3):1349–1362.

[286] Rosenfeld, A. and Wechsler, H. (2000). Pattern recognition: Historical perspective and future directions. *International Journal of Imaging Systems and Technology*, 11(2):101–116.

[287] Sabour, S., Frosst, N., and Hinton, G. E. (2017). Dynamic routing between capsules. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 3856–3866. Curran Associates, Inc.

[288] Sánchez, S., Ramalho, R., Sousa, L., and Plaza, A. (2015). Real-time implementation of remotely sensed hyperspectral image unmixing on gpus. *Journal of Real-Time Image Processing*, 10(3):469–483.

[289] Santos, A. C. S. and Pedrini, H. (2016). A combination of k-means clustering and entropy filtering for band selection and classification in hyperspectral images. *International Journal of Remote Sensing*, 37(13):3005–3020.

[290] Sanz, C. (2001). Der (dynamic evidential reasoning), applied to the classification of hyperspectral images. In *IGARSS 2001. Scanning the Present and Resolving the Future. Proceedings. IEEE 2001 International Geoscience and Remote Sensing Symposium (Cat. No. 01CH37217)*, volume 4, pages 1904–1906. IEEE.

[291] Savage, S. H., Levy, T. E., and Jones, I. W. (2012). Prospects and problems in the use of hyperspectral imagery for archaeological remote sensing: a case study from the faynan copper mining district, jordan. *Journal of Archaeological Science*, 39(2):407 – 420.

[292] Scafutto, R. D. M., de Souza Filho, C. R., and de Oliveira, W. J. (2017). Hyperspectral remote sensing detection of petroleum hydrocarbons in mixtures with mineral substrates: Implications for onshore exploration and monitoring. *ISPRS Journal of Photogrammetry and Remote Sensing*, 128:146 – 157.

[293] Schlapfer, D. R., Kaiser, J. W., Brazile, J., Schaepman, M. E., and Itten, K. I. (2004). Calibration concept for potential optical aberrations of the apex pushbroom imaging spectrometer. In *Sensors, Systems, and Next-Generation Satellites VII*, volume 5234, pages 221–231. International Society for Optics and Photonics.

[294] Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural networks*, 61:85–117.

[295] Schmidt, K. and Skidmore, A. (2003). Spectral discrimination of vegetation types in a coastal wetland. *Remote sensing of Environment*, 85(1):92–108.

[296] Schowengerdt, R. A. (2006). The nature of remote sensing. *Remote Sensing: Models and Methods for Image Processing; Academic Press: New York, NY, USA*, pages 1–44.

[297] Serpico, S. B., D'Inca, M., Melgani, F., and Moser, G. (2003). Comparison of feature reduction techniques for classification of hyperspectral remote-sensing data. In *Image and Signal Processing for Remote Sensing VIII*, volume 4885, pages 347–358. International Society for Optics and Photonics.

[298] Shang, X. and Chisholm, L. A. (2014). Classification of australian native forest species using hyperspectral remote sensing and machine-learning classification algorithms. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 7(6):2481–2489.

[299] Shaw, G. A. and Burke, H. K. (2003). Spectral imaging for remote sensing. *Lincoln laboratory journal*, 14(1):3–28.

[300] Shi, C. and Wang, L. (2014). Incorporating spatial information in spectral unmixing: A review. *Remote Sensing of Environment*, 149:70 – 87.

[301] Shippert, P. et al. (2004). Why use hyperspectral imagery? *Photogrammetric engineering and remote sensing*, 70(4):377–396.

[302] Signoroni, A., Savardi, M., Baronio, A., and Benini, S. (2019). Deep learning meets hyperspectral image analysis: A multidisciplinary review. *Journal of Imaging*, 5(5):52.

[303] Sima, C. and Dougherty, E. R. (2008). The peaking phenomenon in the presence of feature-selection. *Pattern Recognition Letters*, 29(11):1667–1674.

[304] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

[305] Smith, J. S. and Wilamowski, B. M. (2018). Discrete cosine transform spectral pooling layers for convolutional neural networks. In *International Conference on Artificial Intelligence and Soft Computing*, pages 235–246. Springer.

[306] Song, P., Zheng, X., Li, Y., Zhang, K., Huang, J., Li, H., Zhang, H., Liu, L., Wei, C., Mansaray, L. R., et al. (2020). Estimating reed loss caused by *Locusta migratoria manilensis* using uav-based hyperspectral data. *Science of The Total Environment*, 719:137519.

[307] Sonoda, S. and Murata, N. (2017). Neural network with unbounded activation functions is universal approximator. *Applied and Computational Harmonic Analysis*, 43(2):233 – 268.

[308] Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. A. (2014). Striving for simplicity: The all convolutional net. *CoRR*, abs/1412.6806.

[309] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

[310] Srivastava, R. K., Greff, K., and Schmidhuber, J. (2015a). Training very deep networks. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems 28*, pages 2377–2385. Curran Associates, Inc.

[311] Srivastava, R. K., Greff, K., and Schmidhuber, J. (2015b). Training very deep networks. In *Advances in neural information processing systems*, pages 2377–2385.

[312] Stein, D. W. J., Beaven, S. G., Hoff, L. E., Winter, E. M., Schaum, A. P., and Stocker, A. D. (2002). Anomaly detection from hyperspectral imagery. *IEEE Signal Processing Magazine*, 19(1):58–69.

[313] Stirling, W. C. and Morrell, D. R. (1991). Convex bayes decision theory. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(1):173–183.

[314] Strachan, I. B., Pattey, E., and Boisvert, J. B. (2002). Impact of nitrogen and environmental conditions on corn as detected by hyperspectral reflectance. *Remote Sensing of Environment*, 80(2):213 – 224.

[315] Swain, P. H. and Davis, S. M. (1981). Remote sensing: The quantitative approach. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 3(6):713–714.

[316] Tao, D., Jia, G., Yuan, Y., and Zhao, H. (2014). A digital sensor simulator of the pushbroom offner hyperspectral imaging spectrometer. *Sensors*, 14(12):23822–23842.

[317] Tarabalka, Y., Benediktsson, J. A., and Chanussot, J. (2009). Spectral-spatial classification of hyperspectral imagery based on partitional clustering techniques. *IEEE Transactions on Geoscience and Remote Sensing*, 47(8):2973–2987.

[318] Thenkabail, P. S., Lyon, J. G., and Huete, A. (2018). *Hyperspectral Indices and Image Classifications for Agriculture and Vegetation*. CRC Press.

[319] Theodoridis, S. and Koutroumbas, K. (2003). *Pattern Recognition*. Elsevier Science.

[320] Transon, J., d'Andrimont, R., Maugnard, A., and Defourny, P. (2017). Survey of current hyperspectral earth observation applications from space and synergies with sentinel-2. In *2017 9th International Workshop on the Analysis of Multitemporal Remote Sensing Images (MultiTemp)*, pages 1–8.

[321] Vaddi, R. and Prabukumar, M. (2017). Comparative study of feature extraction techniques for hyper spectral remote sensing image classification: a survey. In *2017 International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 543–548. IEEE.

[322] Valero, S., Salembier, P., Chanussot, J., and Cuadras, C. M. (2011). Improved binary partition tree construction for hyperspectral images: Application to object detection. In *2011 IEEE International Geoscience and Remote Sensing Symposium*, pages 2515–2518. IEEE.

[323] Vane, G., Goetz, A. F., and Wellman, J. B. (1984). Airborne imaging spectrometer: A new tool for remote sensing. *IEEE Transactions on Geoscience and Remote Sensing*, GE-22(6):546–549.

[324] Vapnik, V. N. (1999). An overview of statistical learning theory. *IEEE transactions on neural networks*, 10(5):988–999.

[325] Varshney, P. and Arora, M. (2004). *Advanced Image Processing Techniques for Remotely Sensed Hyperspectral Data*. Springer.

[326] Veraverbeke, S., Dennison, P., Gitas, I., Hulley, G., Kalashnikova, O., Katagis, T., Kuai, L., Meng, R., Roberts, D., and Stavros, N. (2018). Hyperspectral remote sensing of fire: State-of-the-art and future perspectives. *Remote Sensing of Environment*, 216:105 – 121.

[327] Von Luxburg, U. and Schölkopf, B. (2011). Statistical learning theory: Models, concepts, and results. In *Handbook of the History of Logic*, volume 10, pages 651–706. Elsevier.

[328] Wang, C., Zhang, P., Zhang, Y., Zhang, L., and Wei, W. (2016). A multi-label hyperspectral image classification method with deep learning features. In *Proceedings of the International Conference on Internet Multimedia Computing and Service*, pages 127–131. ACM.

[329] Wang, F.-Y., Lever, P. J., and Pu, B. (1993). A robotic vision system for object identification and manipulation using synergetic pattern recognition. *Robotics and computer-integrated manufacturing*, 10(6):445–459.

[330] Wang, G. and Weng, Q. (2013). *Remote sensing of natural resources*. CRC Press.

[331] Wang, L., Peng, J., and Sun, W. (2019). Spatial–spectral squeeze-and-excitation residual network for hyperspectral image classification. *Remote Sensing*, 11(7):884.

[332] Wang, Q., Li, Q., Liu, H., Wang, Y., and Zhu, J. (2014). An improved isodata algorithm for hyperspectral image classification. In *2014 7th International Congress on Image and Signal Processing*, pages 660–664.

[333] Wang, W., Dou, S., Jiang, Z., and Sun, L. (2018). A fast dense spectral–spatial convolution network framework for hyperspectral images classification. *Remote Sensing*, 10(7):1068.

[334] Wason, R. (2018). Deep learning: Evolution and expansion. *Cognitive Systems Research*, 52:701–708.

[335] Wei, W., Zhang, L., Tian, C., Plaza, A., and Zhang, Y. (2017). Structured sparse coding-based hyperspectral imagery denoising with intracluster filtering. *IEEE Transactions on Geoscience and Remote Sensing*, 55(12):6860–6876.

[336] Williams, T. and Li, R. (2018). Wavelet pooling for convolutional neural networks. In *International Conference on Learning Representations*, page .

[337] Wold, S., Esbensen, K., and Geladi, P. (1987). Principal Component Analysis. *Chemometrics and Intelligent Laboratory System*, 2(1):37–52.

[338] Wu, H. and Prasad, S. (2018). Semi-supervised deep learning using pseudo labels for hyperspectral image classification. *IEEE Transactions on Image Processing*, 27(3):1259–1270.

[339] Xu, B. and Gong, P. (2008). Noise estimation in a noise-adjusted principal component transformation and hyperspectral image restoration. *Canadian Journal of Remote Sensing*, 34(3):271–286.

[340] Xu, B., Wang, N., Chen, T., and Li, M. (2015). Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*.

[341] Xu, X., Lil, f., and Plaza, A. (2016a). Fusion of hyperspectral and LiDAR data using morphological component analysis. In *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 3575–3578.

[342] Xu, Y., Wu, Z., Li, J., Plaza, A., and Wei, Z. (2016b). Anomaly detection in hyperspectral images based on low-rank and sparse representation. *IEEE Transactions on Geoscience and Remote Sensing*, 54(4):1990–2000.

[343] Yang, C., Everitt, J. H., Bradford, J. M., and Murden, D. (2004). Airborne hyperspectral imagery and yield monitor data for mapping cotton yield variability. *Precision Agriculture*, 5(5):445–461.

[344] Yang, J., Zhao, Y., Chan, J. C., and Yi, C. (2016). Hyperspectral image classification using two-channel deep convolutional neural network. In *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 5079–5082.

[345] Yang, S., Jin, H., Wang, M., Ren, Y., and Jiao, L. (2014). Data-driven compressive sampling and learning sparse coding for hyperspectral image classification. *IEEE Geoscience and Remote Sensing Letters*, 11(2):479–483.

[346] Yegnanarayana, B. (2009). *Artificial neural networks*. PHI Learning Pvt. Ltd.

[347] Yi, C., Zhao, Y. Q., and Chan, J. C. W. (2018). Hyperspectral image super-resolution based on spatial and spectral correlation fusion. *IEEE Transactions on Geoscience and Remote Sensing*, 56(7):4165–4177.

[348] Yi, C., Zhao, Y. Q., Yang, J., Chan, J. C. W., and Kong, S. G. (2017). Joint hyperspectral superresolution and unmixing with interactive feedback. *IEEE Transactions on Geoscience and Remote Sensing*, 55(7):3823–3834.

[349] Yu, D., Wang, H., Chen, P., and Wei, Z. (2014). Mixed pooling for convolutional neural networks. In Miao, D., Pedrycz, W., Ślezak, D., Peters, G., Hu, Q., and Wang, R., editors, *Rough Sets and Knowledge Technology*, pages 364–375, Cham. Springer International Publishing.

[350] Yuan, Q., Shen, H., Li, T., Li, Z., Li, S., Jiang, Y., Xu, H., Tan, W., Yang, Q., Wang, J., et al. (2020). Deep learning in environmental remote sensing: Achievements and challenges. *Remote Sensing of Environment*, 241:111716.

[351] Yue, J., Mao, S., and Li, M. (2016). A deep learning framework for hyperspectral image classification using spatial pyramid pooling. *Remote Sensing Letters*, 7(9):875–884.

[352] Yue, J., Zhao, W., Mao, S., and Liu, H. (2015). Spectral-spatial classification of hyperspectral images using deep convolutional neural networks. *Remote Sensing Letters*, 6(6):468–477.

[353] Zabalza, J., Ren, J., Zheng, J., Zhao, H., Qing, C., Yang, Z., Du, P., and Marshall, S. (2016). Novel segmented stacked autoencoder for effective dimensionality reduction and feature extraction in hyperspectral imaging. *Neurocomputing*, 185:1–10.

[354] Zeiler, M. D. and Fergus, R. (2013). Stochastic pooling for regularization of deep convolutional neural networks. *CoRR*, abs/1301.3557.

[355] Zhan, Y., Hu, D., Wang, Y., and Yu, X. (2017). Semisupervised hyperspectral image classification based on generative adversarial networks. *IEEE Geoscience and Remote Sensing Letters*, 15(2):212–216.

[356] Zhang, C. and Liu, Y. (2014). Multicategory angle-based large-margin classification. *Biometrika*, 101(3):625–640.

[357] Zhang, D., Kang, J., Xun, L., and Huang, Y. (2019). Hyperspectral image classification using spatial and edge features based on deep learning. *International Journal of Pattern Recognition and Artificial Intelligence*.

[358] Zhang, H., He, W., Zhang, L., Shen, H., and Yuan, Q. (2014). Hyperspectral image restoration using low-rank matrix recovery. *IEEE Transactions on Geoscience and Remote Sensing*, 52(8):4729–4743.

[359] Zhang, H. and Ma, J. (2018). Hartley spectral pooling for deep learning. *arXiv preprint arXiv:1810.04028*.

[360] Zhang, L. and Du, B. (2012). Recent advances in hyperspectral image processing. *Geo-spatial Information Science*, 15(3):143–156.

[361] Zhang, L., Zhang, L., and Du, B. (2016). Deep learning for remote sensing data: A technical tutorial on the state of the art. *IEEE Geoscience and Remote Sensing Magazine*, 4(2):22–40.

[362] Zhang, L., Zhang, L., Tao, D., and Huang, X. (2012). On combining multiple features for hyperspectral remote sensing image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 50(3):879–893.

[363] Zhang, L., Zhang, L., Tao, D., Huang, X., and Du, B. (2013). Hyperspectral remote sensing image subpixel target detection based on supervised metric learning. *IEEE transactions on geoscience and remote sensing*, 52(8):4955–4965.

[364] Zhang, L., Zhang, Y., Yan, H., Gao, Y., and Wei, W. (2018). Salient object detection in hyperspectral imagery using multi-scale spectral-spatial gradient. *Neurocomputing*, 291:215–225.

[365] Zhang, M., Qin, Z., Liu, X., and Ustin, S. L. (2003). Detection of stress in tomatoes induced by late blight disease in california, usa, using hyperspectral remote sensing. *International Journal of Applied Earth Observation and Geoinformation*, 4(4):295–310.

[366] Zhao, R., Luk, W., Niu, X., Shi, H., and Wang, H. (2017). Hardware acceleration for machine learning. In *2017 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 645–650. IEEE.

[367] Zhong, G., Ling, X., and Wang, L.-N. (2019). From shallow feature learning to deep learning: Benefits from the width and depth of deep architectures. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 9(1):e1255.

[368] Zhong, P., Gong, Z., Li, S., and Schönlieb, C.-B. (2017a). Learning to diversify deep belief networks for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(6):3516–3530.

[369] Zhong, Y., Wang, X., Zhao, L., Feng, R., Zhang, L., and Xu, Y. (2016). Blind spectral unmixing based on sparse component analysis for hyperspectral remote sensing imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 119:49 – 63.

[370] Zhong, Z., Li, J., Luo, Z., and Chapman, M. (2017b). Spectral–spatial residual network for hyperspectral image classification: A 3-d deep learning framework. *IEEE Transactions on Geoscience and Remote Sensing*, 56(2):847–858.

[371] Zhou, F., Hang, R., Liu, Q., and Yuan, X. (2019). Hyperspectral image classification using spectral-spatial lstms. *Neurocomputing*, 328:39–47.

[372] Zhu, X. X., Tuia, D., Mou, L., Xia, G.-S., Zhang, L., Xu, F., and Fraundorfer, F. (2017). Deep learning in remote sensing: A comprehensive review and list of resources. *IEEE Geoscience and Remote Sensing Magazine*, 5(4):8–36.

[373] Žížala, D., Zádorová, T., and Kapička, J. (2017). Assessment of soil degradation by erosion based on analysis of soil properties using aerial hyperspectral images and ancillary data, czech republic. *Remote Sensing*, 9(1):28.

[374] Zomer, R. J., Trabucco, A., and Ustin, S. (2009). Building spectral libraries for wetlands land cover classification and hyperspectral remote sensing. *Journal of environmental management*, 90(7):2170–2177.

Thesis publications

The following publications have been achieved in the context of this thesis work. Journal papers show the number of citations at May 15, 2020.

Journal Papers:

1. **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, vol. 145, Part A, pp. 120-147, November 2018. DOI: 10.1016/j.isprsjprs.2017.11.021. [IF(2018)=6.942]. Google Scholar Citations: 146. **ESI Highly Cited Paper**.

2. **M. E. Paoletti**, J. M. Haut, R. Fernandez-Beltran, J. Plaza, A. Plaza, J. Li and F. Pla. Capsule Networks for Hyperspectral Image Classification. IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 4, pp. 2145-2160, April 2019. DOI: 10.1109/TGRS.2018.2871782. [IF(2018)=5.630]. Google Scholar Citations: 51. **ESI Research Front**.

3. J. M. Haut, **M. E. Paoletti**, J. Plaza, A. Plaza and J. Li. Visual Attention-Driven Hyperspectral Image Classification. IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 10, pp. 8065-8080, October 2019. DOI: 10.1109/TGRS.2019.2918080. [IF(2018)=5.630]. Google Scholar Citations: 10.

4. **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza. Neural Ordinary Differential Equations for Hyperspectral Image Classification. IEEE Transactions on Geoscience and Remote Sensing, vol. 58, no. 3, pp. 1718-1734, March 2020.
DOI: 10.1109/TGRS.2019.2948031. [IF(2018)=5.630]. Google Scholar Citations: 0.

5. **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza. Neighboring Region Dropout for Hyperspectral Image Classification. IEEE Geoscience and Remote Sensing Letters, accepted for publication, 2020. DOI: 10.1109/LGRS.2019.2940467. [IF(2018)=3.534]. Google Scholar Citations: 0.

6. **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza. Scalable Recurrent Neural Network for Hyperspectral Image Classification. Journal of Supercomputing, accepted for publication, 2020. DOI: 10.1007/s11227-020-03187-0. [IF(2018)=2.157]. Google Scholar Citations: 1.

7. **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza. Deep Learning Classifiers for Hyperspectral Imaging: A Review. ISPRS Journal of Photogrammetry and Remote Sensing, vol. 158, pp. 279-317, December 2019. DOI: 10.1016/j.isprsjprs.2019.09.006. [IF(2018)=6.942]. Google Scholar Citations: 13.

8. J. M. Haut, **M. E. Paoletti**, J. Plaza and A. Plaza. Cloud Implementation of the K-Means Algorithm for Hyperspectral Image Analysis. Journal of Supercomputing, vol. 73, no. 1, pp. 514-529, January, 2017. DOI: 10.1007/s11227-016-1896-3. [IF(2017)=1.532]. Google Scholar Citations: 51.

9. 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, vol. 56, no. 11, pp. 6440-6461, November 2018. DOI: 10.1109/TGRS.2018.2838665. [IF(2018)=5.630]. Google Scholar Citations: 52.

10. 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, vol. 15, no. 3, pp. 439-462, October 2018. DOI: 10.1007/s11554-018-0793-9. [IF(2018)=2.588]. Google Scholar Citations: 14.

11. J. M. Haut, R. Fernandez-Beltran, **M. E. Paoletti**, J. Plaza, A. Plaza and F. Pla. A New Deep Generative Network for Unsupervised Remote Sensing Single-Image Super-Resolution. IEEE Transactions on Geoscience and Remote Sensing, vol. 56, no. 11, pp. 6792-6810, November 2018. DOI: 10.1109/TGRS.2018.2843525. [IF(2018)=5.630]. Google Scholar Citations: 40.

12. R. Fernandez-Beltran, J. M. Haut, **M. E. Paoletti**, J. Plaza, A. Plaza and F. Pla. Multimodal Probabilistic Latent Semantic Analysis for Sentinel-1 and Sentinel-2 Image Fusion. IEEE Geoscience and Remote Sensing Letters, vol. 15, no. 9, pp. 1347-1351, September 2018. DOI: 10.1109/LGRS.2018.2843886. [IF(2018)=3.534]. Google Scholar Citations: 7.

13. N. He, **M. E. Paoletti**, J. M. Haut, L. Fang, S. Li, A. Plaza and J. Plaza. Feature Extraction with Multiscale Covariance Maps for Hyperspectral Image Classification.

IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 2, pp. 755-769, February 2019. DOI: 10.1109/TGRS.2018.2860464. [IF(2018)=5.630].Google Scholar Citations: 48. **ESI Research Front**.

14. **M. E. Paoletti**, J. M. Haut, R. Fernandez-Beltran, J. Plaza, A. Plaza and F. Pla. Deep Pyramidal Residual Networks for Spectral-Spatial Hyperspectral Image Classification. IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 2, pp. 740-754, February 2019 [IF(2018)=5.630]. Google Scholar Citations: 38. **ESI Research Front**.

15. **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza. Deep&Dense Convolutional Neural Network for Hyperspectral Image Classification. Remote Sensing, vol. 10, no. 9, article number 1454, September 2018. [IF(2017)=4.118]. Google Scholar Citations: 24.

16. J. M. Haut, S. Bernabe, **M. E. Paoletti**, R. Fernandez-Beltran, A. Plaza and J. Plaza. Low-High Power Consumption Architectures for Deep Learning Models Applied to Hyperspectral Image Classification. IEEE Geoscience and Remote Sensing Letters, accepted for publication, 2019. DOI: 10.1109/LGRS.2018.2881045. [IF(2018)=3.534]. Google Scholar Citations: 7.

17. R. Fernandez-Beltran, J. M. Haut, **M. E. Paoletti**, J. Plaza, A. Plaza and F. Pla. Remote Sensing Image Fusion Using Hierarchical Multi-Modal Probabilistic Latent Semantic Analysis. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 11, no. 12, pp. 4982-4993, December 2018. DOI: 10.1109/JSTARS.2018.2881342. [IF(2018)=3.392]. Google Scholar Citations: 15.

18. J. M. Haut, **M. E. Paoletti**, J. Plaza, A. Plaza and J. Li. Hyperspectral Image Classification Using Random Occlusion Data Augmentation. IEEE Geoscience and Remote Sensing Letters, accepted for publication, 2019. DOI: 10.1109/LGRS.2019.2909495. [IF(2018)=3.534]. Google Scholar Citations: 6.

19. J. M. Haut, **M. E. Paoletti**, R. Fernandez-Beltran, J. Plaza, A. Plaza and J. Li. Remote Sensing Single-Image Super-Resolution Based on a Deep Compendium Model. IEEE Geoscience and Remote Sensing Letters, accepted for publication, 2019. DOI: 10.1109/LGRS.2019.2899576. [IF(2018)=3.534]. Google Scholar Citations: 3.

20. J. M. Haut, R. Fernandez-Beltran, **M. E. Paoletti**, J. Plaza and A. Plaza. Remote Sensing Image Super-Resolution Using Deep Residual Channel Attention. IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 11, pp. 9277-9289,

November 2019. DOI: 10.1109/TGRS.2019.2924818. [IF(2018)=5.630]. Google Scholar Citations: 5.

21. J. M. Haut, Jose A. Gallardo, **M. E. Paoletti**, G. Cavallaro, J. Plaza, A. Plaza and M. Riedel. Cloud Deep Networks for Hyperspectral Image Analysis. IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 12, pp. 9832-9848, December 2019. DOI: 10.1109/TGRS.2019.2929731. [IF(2018)=5.630]. Google Scholar Citations: 2.

22. J. A. Gallardo, **M. E. Paoletti**, J. M. Haut, R. Fernandez-Beltran, J. Plaza and A. Plaza. GPU Parallel Implementation of Dual-Depth Sparse Probabilistic Latent Semantic Analysis for Hyperspectral Unmixing. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 12, no. 9, pp. 3156-3167, September 2019. DOI: 10.1109/JSTARS.2019.2934011. [IF(2018)=3.392]. Google Scholar Citations: 1.

23. A. Maffei, J. M. Haut, **M. E. Paoletti**, J. Plaza, L. Bruzzone and A. Plaza. A Single Model CNN for Hyperspectral Image Denoising. IEEE Transactions on Geoscience and Remote Sensing, vol. 58, no. 4, pp. 2516-2529, April 2020. DOI: 10.1109/TGRS.2019.2952062. [IF(2018)=5.630]. Google Scholar Citations: 0.

24. A. Alcolea, **M. E. Paoletti**, J. M. Haut, J. Resano and A. Plaza. Inference in Supervised Spectral Classifiers for On-Board Hyperspectral Imaging: An Overview. Remote Sensing, vol. 12, no. 2, 534, Feburary 2020. DOI: 10.3390/rs12030534. [IF(2018)=4.118]. Google Scholar Citations: 0.

25. **M. E. Paoletti**, J. M. Haut, X. Tao, J. Plaza and A. Plaza. GPU Implementation of Support Vector Machines for fast Remotely Sensed Hyperspectral data classification. Remote Sensing, vol. 12, no. 8, 1257, April 2020. DOI: 10.3390/rs12081257. [IF(2018)=4.118]. Google Scholar Citations: 0. **Selected as journal cover**.

26. S. Moreno-Álvarez, J. M Haut, **M. E Paoletti**, J. A. Rico-Gallego, J. C. Díaz-Martín, J. Plaza. Training deep neural networks: a static load balancing approach. Journal of Supercomputing, accepted for publication, 2020. DOI: 10.1007/s11227-020-03200-6. [IF(2018)=2.157]. Google Scholar Citations: 0.

27. T. Alipour-Fard, **M. E. Paoletti**, J. M. Haut, H. Arefi, J. Plaza and A. Plaza. Multi-branch Selective Kernel Networks for Hyperspectral Image Classification. IEEE Geoscience and Remote Sensing Letters, accepted for publication, 2020. DOI: 10.1109/LGRS.2020.2990971. [IF(2018)=3.534]. Google Scholar Citations: 0.

Submited Journal Papers:

1. **M. E. Paoletti**, J. M. Haut, X. Tao, J. Plaza and A. Plaza. Parameter-free Convolutional Neural Network for Hyperspectral Image Classification. IEEE Transactions on Geoscience and Remote Sensing. [IF(2018)=5.630].

2. X. Tao, **M. E. Paoletti**, J. M. Haut, P. Ren, J. Plaza and A. Plaza. Endmember Estimation with Maximum Distance Analysis. IEEE Transactions on Geoscience and Remote Sensing. [IF(2018)=5.630].

3. **M. E. Paoletti**, J. M. Haut, P. Ghamisi, N. Yokoya, J. Plaza and A. Plaza. U-IMG2DSM: Unpaired Simulation of Digital Surface Models with Generative Adversarial Networks. IEEE Geoscience and Remote Sensing Letters. [IF(2018)=3.534].

Journal Papers in Spanish:

1. **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza. Estudio Comparativo de Técnicas de Clasificación de Imágenes Hiperespectrales. Revista Iberoamericana de Automática e Informática industrial, 16(2), 129 - 137. 2019. DOI: 10.4995/riai.2019.11078. [IF(2018)=1.313]. Google Scholar Citations: 2.

Peer-Reviewed International Conference Papers:

1. J. M. Haut, **M. E. Paoletti**, J. Plaza and A. Plaza. Cloud Implementation of the K-Means Algorithm for Hyperspectral Image Analysis. Proceedings of the International Conference on Computational and Mathematical Methods in Science and Engineering (CMMSE'16), Rota, Spain, 2016. ISBN: 978-84-608-608-6082-2.

2. J. M. Haut, **M. E. Paoletti**, A. Paz-Gallardo, J. Plaza and A. Plaza. Cloud Implementation of the Logistic Regression for Hyperspectral Image Analysis. Proceedings of the International Conference on Computational and Mathematical Methods in Science and Engineering (CMMSE'17), Rota, Spain, 2017. ISBN: 978-84-617-8694-7.

3. **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza. Yinyang K-means clustering for hyperspectral image analysis. Proceedings of the International Conference on Computational and Mathematical Methods in Science and Engineering (CMMSE'17), Rota, Spain, 2017. ISBN: 978-84-617-8694-7.

4. M. Peñalver, F. Del Frate, **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza. Onboard Payload-Data Dimensionality Reduction. IEEE Geoscience and Remote Sensing Symposium (IGARSS'17), Fort Worth, Texas, 2017. DOI: 10.1109/IGARSS.2017.8127069.

5. **M. E. Paoletti**, J. M. Haut, J. Plaza, A. Plaza, Q. Liu and R. Hang. Multicore Implementation of the Multi-Scale Adaptive Deep Pyramid Matching Model for Remotely Sensed Image Classification. IEEE Geoscience and Remote Sensing Symposium (IGARSS'17), Fort Worth, Texas, 2017. DOI: 10.1109/IGARSS.2017.8127436.

6. J. M. Haut, Y. Liu, **M. E. Paoletti**, X. Xu, J. Plaza and A. Plaza. Evaluation of Different Regularization Methods for the Extreme Learning Machine Applied to Hyperspectral Images. IEEE Geoscience and Remote Sensing Symposium (IGARSS'18), Valencia, Spain, 2018. DOI: 10.1109/IGARSS.2018.8518746.

7. J. M. Haut, R. Fernandez-Beltran, **M. E. Paoletti**, J. Plaza, A. Plaza and F. Pla. Inter-Sensor Regression Analysis for Operational Sentinel-2 and Sentinel-3 Data Products. IEEE Geoscience and Remote Sensing Symposium (IGARSS'18), Valencia, Spain, 2018. DOI: 10.1109/IGARSS.2018.8517976.

8. **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza. An Investigation of Self-Normalized Deep Neural Networks for Hyperspectral Image Classification. IEEE Geoscience and Remote Sensing Symposium (IGARSS'18), Valencia, Spain, 2018. DOI: 10.1109/IGARSS.2018.8517449.

9. J. M. Haut, **M. E. Paoletti**, J. Plaza, A. Plaza and F. del Frate. Cloud Computing Implementation of a Neural Classifier for Remotely Sensed Hyperspectral Image. MED2018. European Space Research Institute of European Space Agency (ESRIN-ESA), Frascati, Italy, 2018.

10. C. Liu, J. Li, **M. E. Paoletti**, J. M. Haut, A. Plaza, Q. Shi. Accessibility-Free Active Learning for Hyperspectral Image Classification. IEEE Geoscience and Remote Sensing Symposium (IGARSS'19), Yokohama, Japan, 2019.

11. S. Bernabé García, C. García, R. Fernandez-Beltrán, **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza. Open Multi-Processing Acceleration For Unsupervised Land Cover Categorization Using Probabilistic Latent Semantic Analysis. IEEE Geoscience and Remote Sensing Symposium (IGARSS'19), Yokohama, Japan, 2019.

12. **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza. Solving Deep Neural Networks with Ordinary Differential Equations for Remotely Sensed Hyperspectral Image Classifica-

tion. IEEE Geoscience and Remote Sensing Symposium (IGARSS'19), Yokohama, Japan, 2019.

13. **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza. Scalable Recurrent Neural Network for Spectral-Spatial Classification of Hyperspectral Images. Proceedings of the International Conference on Computational and Mathematical Methods in Science and Engineering (CMMSE'19), Rota, Spain, 2019.

14. S. Moreno-Álvarez, **M. E. Paoletti**, J. M. Haut, J. A. Rico-Gallego, J. Plaza, J C. Diaz-Martin. Exploring Distributed Deep Network Training Accuracy and Performance on Heterogeneous Clusters. Proceedings of the International Conference on Computational and Mathematical Methods in Science and Engineering (CMMSE'19), Rota, Spain, 2019.

15. A. Maffei, J. M. Haut, **M. E. Paoletti**, J. Plaza, L. Bruzzone and A. Plaza. Efficient Convolutional Neural Network for Spectral-Spatial Hyperspectral Denoising. IEEE Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS'19), Amsterdam, The Netherlands, 2019

Accepted for Presentation Peer-Reviewed International Conference Papers:

1. **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza. Training CapsNets via Active Learning for Hyperspectral Image classification. IEEE Geoscience and Remote Sensing Symposium (IGARSS'20), Waikoloa, EEUU, 2020.

National Conferences:

1. J. M. Haut, **M. E. Paoletti**, J. Plaza and A. Plaza. Evaluación de rendimiento de una implementación Cloud para un clasificador neuronal aplicado a imágenes hiperespectrales. Jornadas SARTECO Congreso Español de Informática (CEDI'2016), Salamanca, Spain, 2016.

2. J. M. Haut, **M. E. Paoletti**, J. A. Gallardo, J. Plaza and A. Plaza. Red neuronal profunda distribuida para compresión de imágenes hiperspectrales. Jornadas SARTECO, Teruel, Spain, 2018.

3. J. A. Gallardo, J. M. Haut, **M. E. Paoletti**, A. Plaza and J. Plaza. Nueva implementación paralela en GPUs del algoritmo pLSA para desmezclado de imágenes hiperespectrales. Jornadas SARTECO, Cáceres, Spain, 2019.

4. M. Blanco, J. M. Haut, **M. E. Paoletti**, A. Plaza and J. Plaza. Una nueva plataforma social para el procesamiento de imágenes hiperespectrales de manera masiva. Jornadas SARTECO, Cáceres, Spain, 2019.

5. S. Moreno, **M. E. Paoletti**, J. M. Haut, J. A. Rico-Gallego, J. Plaza and J.C. Díaz-Martín. Evaluación de Rendimiento del Entrenamiento Distribuido de Redes Neuronales Profundas en Plataformas Heterogéneas. Jornadas SARTECO, Cáceres, Spain, 2019.

6. A. Maffei, J. M. Haut, **M. E. Paoletti**, J. Plaza, L. Bruzzone and A. Plaza. Una única arquitectura neuronal profunda para para eliminar ruido en imágenes hiperespectrales. Jornadas SARTECO, Cáceres, Spain, 2019.

Copyright notice

Escuela Politecnica
Av. de la Universidad, S/N, 10003
Caceres, Spain
Phone: 0034927257000. Ext. 51662
Email: aplaza,jplaza{@unex.es}

Dr. Antonio Plaza Miguel y Dr. Javier Plaza Miguel como directores de la tesis titulada "Procesamiento eficiente y profundo de imgenes hiperespectrales de la observación remota de la Tierra y aplicaciones en tareas de clasificación", certifico el factor de impacto y la categorización de la siguiente publicación, incluida en la tesis doctoral. Del mismo modo, se especifica la aportación del doctorado. Si necesitas cualquier información o clarificación, por favor, no dude en contactar conmigo.

Antonio Plaza Miguel PhD. and Javier Plaza Miguel PhD as directors of the Phd thesis titled "Deep-efficient processing of remote sensing hyperspectral images and applications in classification tasks.", certify the impact factor and the categorization of the following publication, included in the doctoral thesis. In the same way, the contribution of the doctorate is specified. If you need any further information or clarification, please do not hesitate contacting me.

Artculo / Paper

Autores/Authors: **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza.
Title: Deep Learning Classifiers for Hyperspectral Imaging: A Review.
Journal: ISPRS Journal of Photogrammetry and Remote Sensing.
Other Information: vol. 158, pp. 279-317, December 2019.
DOI: 10.1016/j.isprsjprs.2019.09.006.
Impact factor 2018: 6.942. Q1
Abstract: Advances in computing technology have fostered the development of new and powerful deep learning (DL) techniques, which have demonstrated promising results in a wide range of applications. Particularly, DL methods have been successfully used to classify remotely sensed data collected by Earth Observation (EO) instruments. Hyperspectral imaging (HSI) is a hot topic in remote sensing data analysis due to the vast amount of information comprised by this kind of images, which allows for a better characterization and exploitation of the Earth surface by combining rich spectral and spatial information. However, HSI poses major challenges for supervised classification methods due to the high dimensionality of the data and the limited availability of training samples. These issues, together with the high intraclass variability (and interclass similarity) often present in HSI data may hamper the effectiveness of classifiers. In order to solve these limitations, several DL-based architectures have been recently developed, exhibiting great potential in HSI data interpretation. This paper provides a comprehensive review of the current-state-of-the-art in DL for HSI classification, analyzing the strengths and weaknesses of the most widely used classifiers in the literature. For each discussed method, we provide quantitative results using several well-known and widely used HSI scenes, thus providing an exhaustive comparison of the discussed techniques. The paper concludes with some remarks and hints about future challenges in the application of DL techniques to HSI classification.

Contribución del doctorado: Planteamiento de la hipótesis, desarrollo práctico, análisis y discusión de los resultados, elaboración y escritura del manuscrito.

Firma / Signature
Mayo / May, 2020

Antonio Plaza Miguel

Javier Plaza Miguel

ISPRS Journal of Photogrammetry and Remote Sensing 158 (2019) 279–317

Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

journal homepage: www.elsevier.com/locate/isprsjprs

Deep learning classifiers for hyperspectral imaging: A review

M.E. Paoletti*, J.M. Haut, J. Plaza, A. Plaza

Hyperspectral Computing Laboratory (HyperComp), Department of Computer Technology and Communications, Escuela Politecnica de Caceres, University of Extremadura, Avenida de la Universidad s/n, E-10003 Caceres, Spain

ARTICLE INFO

Keywords:
Deep learning (DL)
Hyperspectral imaging (HSI)
Earth observation (EO)
Classification

ABSTRACT

Advances in computing technology have fostered the development of new and powerful deep learning (DL) techniques, which have demonstrated promising results in a wide range of applications. Particularly, DL methods have been successfully used to classify remotely sensed data collected by Earth Observation (EO) instruments. Hyperspectral imaging (HSI) is a hot topic in remote sensing data analysis due to the vast amount of information comprised by this kind of images, which allows for a better characterization and exploitation of the Earth surface by combining rich spectral and spatial information. However, HSI poses major challenges for supervised classification methods due to the high dimensionality of the data and the limited availability of training samples. These issues, together with the high intraclass variability (and interclass similarity) –often present in HSI data– may hamper the effectiveness of classifiers. In order to solve these limitations, several DL-based architectures have been recently developed, exhibiting great potential in HSI data interpretation. This paper provides a comprehensive review of the current-state-of-the-art in DL for HSI classification, analyzing the strengths and weaknesses of the most widely used classifiers in the literature. For each discussed method, we provide quantitative results using several well-known and widely used HSI scenes, thus providing an exhaustive comparison of the discussed techniques. The paper concludes with some remarks and hints about future challenges in the application of DL techniques to HSI classification. The source codes of the methods discussed in this paper are available from: https://github.com/mhaut/hyperspectral_deeplearning_review.

1. Introduction

Imaging spectroscopy, also called hyperspectral imaging (HSI), studies how the light interacts with the observed materials, measuring the amount of light that is emitted, reflected or transmitted from a certain object or target. Imaging spectrometers (also called HSI sensors) usually operate in the 0.4 to 2.5 μm spectral region, capturing the visible and solar-reflected infrared spectrum (i.e., the near-infrared or NIR, and the short-wavelength infrared or SWIR) from the observed materials. However, as opposed to broad-band sensing systems that under-sample the available spectral information, narrow-band HSI systems are able to produce, for each captured target, a distinctive spectral signature composed by reflectance measurements at hundreds of different wavelength channels (Goetz et al., 1985). The exploitation of spectral signatures as unique *fingerprints* makes imaging spectrometry an interesting and powerful tool for the categorization of the surface of the Earth, reaching promising results in a wide range of applications (Huadong et al., 2001; Transon et al., 2017; Khan et al., 2018; Transon et al., 2018). In the current literature, a great number of works focus on the use of HSI data for resource management. For instance, in agricultural applications (Teke et al., 2013) there are several works focused on the analysis of environmental stress in crops and associated diseases (Strachan et al., 2002; Feng et al., 2017), crops variability (Yang et al., 2004; Rußwurm and Körner, 2017), soil erosion stages (Bannari et al., 2006; Chabrillat et al., 2014) or precision agriculture (Haboudane et al., 2004; Rodríguez-Pérez et al., 2007; Mahesh et al., 2015), among many others. In forestry and environmental management, relevant works have been presented on analyzing the status and health of forests (Coops et al., 2003; Shang and Chisholm, 2014), invasive species detection (Ustin et al., 2002a; Große-Stoltenberg et al., 2016), and infestations in plantation forestry (Narumalani et al., 2009; Peerbhay et al., 2015). Also, in water and maritime resources management (Younos and Parece, 2015), several studies have focused on water quality analysis (Koponen et al., 2002; Olmanson et al., 2013; El-Magd and El-Zeiny, 2014) and precipitations (Zhou et al., 2011) or sea ice detection (Han et al., 2017). In geological exploration and mineralogy, HSI data have been used for detection and mapping of mineral deposits (Resmini et al., 1997; Kokaly et al., 2013; Kokaly et al., 2016; Mazhari et al., 2017; Scafutto et al., 2017; Aslett et al., 2018; Dumke et al., 2018; Acosta et al., 2019) or soil composition analysis (Shi et al.,

* Corresponding author.
E-mail addresses: mpaoletti@unex.es (M.E. Paoletti), juanmariohaut@unex.es (J.M. Haut), jplaza@unex.es (J. Plaza), aplaza@unex.es (A. Plaza).

https://doi.org/10.1016/j.isprsjprs.2019.09.006
Received 6 November 2018; Received in revised form 9 September 2019; Accepted 9 September 2019
Available online 19 November 2019

2014). Other areas in which the use of HSI provided relevant results include urban planning (Abbate et al., 2003; Lulla et al., 2009; Heldens et al., 2011; Man et al., 2015; Anand et al., 2017), disaster prediction (Ustin et al., 2002b; Roberts et al., 2003; Transon et al., 2018; Veraverbeke et al., 2018), military and defense applications (Richter, 2005; Briottet et al., 2006; Ardouin et al., 2007; El-Sharkawy and Elbasuney, 2019) and archaeological analyses (Savage et al., 2012).

Several efforts have been made over the past decades to produce high-quality HSI data for Earth Observation (EO) (Lucas et al., 2004; Ghamisi et al., 2017b), developing a wide range of imaging spectrometers placed on aerial/satellite platforms, and recently also on stationary or hand-held platforms. These sensors combine the power of digital imaging and spectroscopy to extract, for every location in an image plane, the corresponding spectral signature, using thousands of narrow and continuous bands and acquiring complete HSI data cubes by raster-scanning the scene while the platform moves across the surface (i.e. pushbroom sensors), covering large observation areas. As result, the captured area or scene is recorded in different wavebands, creating a huge data cube $\mathbf{X} \in \mathbb{N}^{n_1 \times n_2 \times n_{bands}}$, composed by $(n_1 \times n_2)$ spectral vectors or HSI pixels, where each $\mathbf{x}_i \in \mathbb{N}^{n_{bands}}$ records the spectral signature of the observed material.

Nowadays, several instruments are routinely capturing great volumes of HSI data, with some of them exhibiting high acquisition rates, i.e. being able to capture gigabytes (GBs) or even terabytes (TBs) of data per hour (Vane et al., 1989; Kruse et al., 2000). In this regard, Table 1 provides the specifications of some of the best-known spectrometers currently available. Moreover, advances in computing technologies have achieved great improvements in the data acquisition, storage and processing procedures, allowing also the launch of a number of HSI-EO missions –such as the NASA Hyperspectral Infrared Imager (HyspIRI) (Roberts et al., 2012), the Environmental Mapping and Analysis Program (EnMAP) (Kaufmann et al., 2008) or the Precursore IperSpettrale della Missione Applicativa (PRISMA) program (Galeazzi et al., 2008)– as well as the practical application of remotely sensed HSI data in real scenarios (Tuia and Camps-Valls, 2009; Zhang and Du, 2012), providing a general idea about the importance and utility of HSI-based remote sensing.

The specialized literature about remotely sensed HSI data covers a wide range of processing techniques that can efficiently extract the information contained in the HSI cube. The most popular ones include: (i) spectral unmixing (Bioucas-Dias et al., 2012; Heylen et al., 2014; Shi and Wang, 2014; Sánchez et al., 2015; Zhong et al., 2016a), (ii) resolution enhancement (Eismann and Hardie, 2005; Mookambiga and Gomathi, 2016; Yi et al., 2017; Yi et al., 2018), (iii) image restoration and denoising (Xu and Gong, 2008; Chen and Qian, 2011; Zhang et al., 2014; Wei et al., 2017b), (iv) anomaly detection (Stein et al., 2002; Xu et al., 2016; Kang et al., 2017), (v) dimensionality reduction (Bruce et al., 2002; Haut et al., 2018d) and (vi) data classification (Fauvel et al., 2013; Camps-Valls et al., 2014; Ghamisi et al., 2017a). In this work, we particularly focus on the topic of HSI data classification, which has received remarkable attention due its important role in land use and land cover applications (Cheng et al., 2017a), and which is currently one of the most popular techniques for HSI data exploitation (Chang, 2007).

A wide variety of HSI data classification methodologies rely on machine learning (ML) techniques (Kotsiantis et al., 2006; Kotsiantis et al., 2007), which are already collected in an extensive list of detailed reviews, such as Plaza et al. (2009), Zhang and Du (2012), Ablin and Sulochana (2013), Fauvel et al. (2013), Camps-Valls et al. (2014), Li and Du (2016), Chutia et al. (2016), Ghamisi et al. (2017b), Chen et al. (2014b), or even more recently in Li et al. (2019a), Audebert et al. (2019), Signoroni et al. (2019), among others. However, ML is a field in constant evolution, where new and improved methods are designed from time to time. In this sense, from the early 2000s, the ML field has experimented a significant revolution thanks to the development of new deep learning (DL) models (Schmidhuber, 2015), which have been supported by advances in computer technology. These models have become an inspiration for the development of new and improved HSI data classifiers, marking a clear trend since 2017 (Petersson et al., 2016; Ghamisi et al., 2017a; Zhu et al., 2017). In this sense, the aim of this work is to delve deeper into those classification techniques based on DL techniques, providing an updated review about the most popular models and widely used architectures to perform remotely sensed HSI data classification.

The remainder of the paper is organized as follows. In Section 2, we introduce the problem of HSI data classification, providing a brief framework for ML and DL methods, introducing the general benefits of DL models and their limitations, coupled with the challenges that must be faced when working with remotely sensed HSI data. Section 3 introduces some general DL concepts, while Section 4 reviews the principal DNN architectures employed for HSI data classification. Section 5 introduces some widely-used techniques to overcome DL and HSI limitations. Section 6 presents some popular programming frameworks for the development of DL models. Section 7 provides an experimental evaluation of the discussed methods using several well-known HSI data sets. Our experimental assessment includes a detailed discussion of the results obtained in terms of accuracy and performance. Section 8 concludes the paper with a discussion on future trends, including ongoing computational developments such as the use of parallelization and distribution techniques via graphical processing units (GPUs) and cloud computing environments.

Table 1
Some of the most widely-known HSI sensors, highlighting several of their spectral-spatial characteristics. In particular, we outline the spectral features, the number of bands, range (μm), and spectral resolution (nm), taking into account also the spatial ground sample distance measured in meters per pixel (mpp).

	Sensor	Bands	Range	Width	GSD
Airborne	AVIRIS (Green et al., 1998)	224	0.36–2.45	10	20
	AVIRIS-NG (Bue et al., 2015)	600	0.38–2.51	5	0.3–4.0
	CASI (Babey and Anger, 1989)	144	0.36–1.05	2.4	2.5
	HYDICE (Rickard et al., 1993)	210	0.40–2.50	10.2	1–7
	HYMAP (Cocks et al., 1998)	126	0.45–2.50	15	5
	PRISM (Mouroulis et al., 2014)	248	0.35–1.05	3.5	2.5
	ROSIS (Kunkel et al., 1988)	115	0.43–0.86	4	1.3
Satellite	EnMAP (Guanter et al., 2015)	228	0.42–2.40	5.25–12.5	30
	DESIS (Eckardt et al., 2015)	180	0.40–1.00	3.30	30
	HYPERION (Pearlman et al., 2003)	220	0.40–2.50	10	30
	PRISMA (Pignatti et al., 2013)	237	0.40–2.50	≤12	30
	SHALOM (Feingersh and Dor, 2015)	241	0.40–2.50	10	10

2. Hyperspectral data classification: backgrounds and challenges

2.1. From traditional machine learning methods to deep learning models

Any classification problem can be mathematically formulated as an optimization one, where a mapping function (with or without certain parameters θ) $f_c(\cdot,\theta)$ receives an input data sample $\mathcal{X}$ and obtains the corresponding label category, $\mathcal{Y}$, by applying several transformations over the original input, i.e. $f_c: \mathcal{X} \rightarrow \mathcal{Y}$, with the aim of minimizing the gap between the desired output and the obtained one. In this regard, the purpose of classifying HSI data is to categorize those pixels $\mathbf{x}_i \in \mathbb{N}^{n_{bands}}$ (spectral vectors) contained in the HSI scene $\mathbf{X} \in \mathbb{N}^{n_1 \times n_2 \times n_{bands}}$ into a set of unique and mutually exclusive land cover classes (He et al., 2017a), obtaining the classification map $\mathbf{Y} \in \mathbb{N}^{n_1 \times n_2} \subset \{1, ..., n_{classes}\}$. Moreover, it is usual to binarize each category, performing the so-called one-hot encoding $\mathbf{Y} \in \mathbb{N}^{n_1 \times n_2 \times n_{classes}}$, so the mapping function $\mathbf{Y} = f_c(\mathbf{X}, \theta)$ assigns a vector label $\mathbf{y}_i \in \mathbb{N}^{n_{classes}}$ to each spectral pixel, $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{(n_1 \cdot n_2)}$.

In the literature, there is a vast amount of works about HSI data classification. Usually, these methods have been inspired by those algorithms and techniques developed in the fields of computer vision and pattern recognition, exhibiting a wide variety of methodologies and learning procedures. As a result, they can be divided in many groups depending on multiple factors, from unsupervised methods (for instance: k-means (Haut et al., 2017b), k-nearest neighbors -KNN- (Cariou and Chehdi, 2015) or iterative self-organizing data analysis technique -ISODATA- (Wang et al., 2014)) to supervised ones (support vector machines -SVMs- (Melgani and Bruzzone, 2004) or random forests -RFs- (Ham et al., 2005)), from statistical classifiers (such as multinomial logistic regression -MLR- (Haut et al., 2017a)) to deterministic methods (for instance, extreme learning machines -ELMs- (Li et al., 2018a)), from parametric algorithms (such as the maximum likelihood -ML- (Kuching, 2007)) to non-parametric ones (such as the evidential reasoning -ER- (Sanz, 2001)), from spectral-based methodologies (traditional distance metrics based classifiers (Du and Chang, 2001; Keshava, 2004), spectral angle mapper -SAM- (Camps-Valls, 2016; Calin et al., 2018), etc.) to spatial or spectral-spatial ones (sparse coding -SC- (Charles et al., 2011; Yang et al., 2014), morphological profiles -MP- (Fauvel et al., 2008; Huang and Zhang, 2013; Bhardwaj and Patra, 2018), among others). In this regard, several taxonomies have been proposed in order to categorize the available methods. For instance, Lu and Weng (2007) offered an interesting and complete taxonomy of thirteen categories, considering six different criteria, while Chutia et al. (2016) presented a simpler taxonomy of six different groups depending on the classification procedure. Also, Ghamisi et al. (2017a) provided a complex taxonomy with eight criteria and twenty categories, although none of them are exclusively dedicated to DL methods.

In fact, DL is a subfield of ML inspired by the structure and functions of the biological brain (Bengio, 2009; LeCun et al., 2015; Goodfellow et al., 2016), so those DL-HSI classifiers are often framed within the field of artificial neural networks (ANNs) (Plaza et al., 2011b), which are characterized by their flexible architecture, composed by groups (layers) of connected computational units (neurons). ANNs work on the basis that the global classification problem defined by f_c is split into several hierarchically ordered sub-mapping functions $\mathbf{Y} = f_c(\mathbf{X}, \theta) \approx \hat{f}\,(f^{(L)}(\cdots(f^{(1)}(\mathbf{X}, \theta^{(1)})\cdots), \theta^{(L)}))$, being L the number of layers that compose the network, $\mathbf{X}$ the original input data and $\hat{f}$ the final classifier (performed by a classification layer in end-to-end models or by any standard ML classifier). This is supported by the assumption that approximating a high number of small steps is better than solving a small number of large steps, implementing a "divide & conquer" strategy. In this context, each $f^{(l)}$ is of the general form defined by Eq. (1):

$$\mathbf{X}^{(l)} = f^{(l)}(\mathbf{X}^{(l-1)}, \mathbf{W}^{(l)}, b^{(l)}), \tag{1}$$

where weights $\mathbf{W}^{(l)}$ and biases $b^{(l)}$ are the parameters $\theta^{(l)}$ of the sub-mapping function $f^{(l)}$, and $\mathbf{X}^{(l-1)}$ and $\mathbf{X}^{(l)}$ are the input and output data, respectively. Moreover, ANNs are inspired by the neural connections that conform the biological brain's structure and the pulses that travel through synaptic connections to transmit information. In this sense, each $f^{(l)}$ is in fact composed by a set of neurons, which apply their corresponding synaptic weights over the input data, and whose responses are filtered, determining the neural activations which will be forwarded to the following $f^{(l+1)}$.

This hierarchical structure of stacked functions has fostered the rise of deep and very deep ANN models, as described in the outstanding and comprehensive overview presented in Zhang et al. (2016b). These models will be referred to hereinafter as DNNs and VDNNs. In this regard, although the limits between one type of network and another have not been established (Schmidhuber, 2015), there is an agreement among the experts to establish a distinction between shallow and deep architectures (Bengio et al., 2007b), whereby single-hidden layer structures are considered as shallow ANNs, architectures with two or more hidden layers are considered as DNNs, and models with dozens of layers are categorized as VDNNs (Srivastava et al., 2015). For instance, Hinton et al. (2006) presented a neural model with three hidden layers as one of the first deep architectures; Krizhevsky et al. (2012) considered their model with more than 5 layers as a deep network, and Simonyan and Zisserman (2014) introduced a VDNN with 16–19 layers. Following this trend, extremely deep neural networks (EDNN, also known as ultra-deep nets) have been introduced as architectures with more than 50 layers, reaching even thousands of layers (He et al., 2016). In this context, the stack of functions allows to extract data representations at different levels, which are processed by the successive neural layers. In fact, any ANN works as a feature extractor (FE), regardless of its depth, where each sub-mapping function encodes different characteristics from the input data. In general, these models' architecture allows for the learning of generic features at the early stages, a piece of knowledge that is traditionally considered as less dependent on the application, while the final layers are able to learn pieces of knowledge that are more related with the application at hand. This allows for the extraction of highly abstract data representations, which are directly obtained and refined by the classification problem itself, being modeled by each $\theta^{(l)}$ of the architecture. This also allows a higher flexibility in comparison with those methods that are subordinated to *hand-crafted features*, which should manually design the desired features, employing some well-known FE methods such as the scale invariant feature transform -SIFT- (Al-khafaji et al., 2018), histogram of oriented gradients -HOG-, local binary patterns -LBP- (Li et al., 2015b), or speeded-up robust features -SURF-, among others. This last procedure imposes several restrictions, in particular, the obtained features are very specific and usually exhibit limited levels of invariance and abstraction. Also, they are critically dependent on the user's knowledge, making hard to guarantee their setup (Yang et al., 2016).

In turn, the structure and functions of ANNs makes them universal approximators (Cybenko, 1989; Hornik, 1991), allowing them to learn any data system's behavior without any prior or additional information about the statistical distribution of the input data. In this sense, ANNs have attracted the attention of a large number of researchers in the area of HSI data classification (Benediktsson et al., 1993; Yang, 1999), and nowadays also in their DL version (Chen and Wang, 2014), due to the benefits that DNN models exhibit when compared to traditional ML methods (Collobert and Bengio, 2004):

1. The ability to extract hidden and sophisticated structures (both, linear and non-linear features) contained in the raw data. Such ability is intrinsically related, on the one hand, to the capacity to model their own internal representation (rather than having it prespecified, as handcrafted features by kernel functions (Camps-Valls et al., 2006)) and, on the other hand, to their ability for generalizing any kind of knowledge.

2. They are extremely flexible in the types of data they can support. In particular, they can take advantage of the spectral and spatial domains of HSI data, in both separate and coupled fashion.
3. Also, they offer a large flexibility on architectures, in terms of the type of layers, blocks or units, and their depth.
4. Moreover, their learning procedure can be adapted to a great variety of learning strategies, from unsupervised to supervided techniques, going through intermediate strategies.
5. Finally, advances in processing techniques such as batch partition and high performance computing (HPC) (Plaza et al., 2009; Lee et al., 2011; Bioucas-Dias et al., 2013), in particular on parallel and distributed architectures (Plaza and Chang, 2008; Plaza et al., 2011a), have allowed DNN models to scale better when dealing with large amounts of data.

These characteristics make DNNs very powerful and popular models for HSI data classification. However, as traditional ML approaches, DNNs are not exempt from certain limitations, which are highly related to the characteristics of HSI data.

2.2. Hyperspectral data challenges and deep learning limitations

ANN classifiers in general (and DL-based models in particular) need to face some challenges related to the processing of high-spectral dimensional data sets such as HSI data cubes. In fact, although the rich spectral information contained in each pixel $\mathbf{x}_i \in \mathbb{N}^{n_{bands}}$ is very useful to perform an accurate discrimination process, its large dimensionality brings new challenges, not only in terms of computation time and storage, but also due to the so-called peaking paradox (Theodoridis and Koutroumbas, 2003; Kallepalli et al., 2014; Sima and Dougherty, 2008). This paradox establishes that the use of additional features (i.e., spectral bands) brings complexity into the classifier, increasing the number of statistical parameters that define the land cover classes, and which must be estimated in advance. Following the previous notation, if we formulate the classification process as the approximation of a function $f_c: \mathbb{N}^{n_{bands}} \rightarrow \mathbb{N}^{n_{classes}}$ that identifies, for each spectral pixel $\mathbf{x}_i \in \mathbf{X}$, its corresponding label vector $f_c(\mathbf{x}_i) = \mathbf{y}_i$,we can infer that the corresponding estimation errors will increase when more parameters/features are taken into account, hampering the final classification performance (Landgrebe, 2005). This leads to the *curse of dimensionality* problem (Bellman, 2015) that greatly affects supervised classification methods, in which the size of the training set may not be sufficient to accurately derive the statistical parameters, thus leading the classifier to quickly overfit (Hughes phenomenon (Hughes, 1968)).

Coupled with their high dimensionality, HSI data presents several artefacts that make the classification process a difficult task. Similar to very high-resolution (VHR) images, HSI data also suffers a high intraclass variability, resulting from uncontrolled changes in the reflectance captured by the spectrometer (normally because of changes in atmospheric conditions, occlusions due to the presence of clouds, and variations in illumination, among other environmental interferers). Also, the instrumental noise produced by the spectrometer may degrade the data acquisition process, corrupting spectral bands to different degrees (Rasti et al., 2018), or even making several bands unusable due to saturation/cutoff or calibration errors (Pearlman et al., 2003). Also, there is a tendency in HSI instruments to include significant redundancy across adjacent spectral bands, which leads to the presence of redundant information that may hinder computational efficiency of analysis algorithms. Regarding the spatial information, pixels in HSI data often cover large spatial regions on the surface of the Earth in images with low/medium spatial resolution, so they tend to generate mixed spectral signatures, leading to high interclass similarity in border regions. In the end, these challenges create potential ambiguities and uncertainties (Varshney and Arora, 2004; Gomez et al., 2015) that must be faced by classification algorithms in order to extract representative features from the images.

Another important issue is the problematic lack of labelled data. In fact, despite the launch and start-up of the HSI-EO missions described on Section 1, the number of operational spaceborne spectrometers that are continuously acquiring images is still low in comparison with multispectral remote sensing sensors such as Landsat or the Sentinel missions, and in general the captured data are not publicly offered. Moreover, airborne spectrometers cover much smaller areas than those sensors allocated on satellite platforms, so the amount of HSI datasets is quite limited. In addition, the task of labelling each pixel contained in the HSI dataset is arduous and time-consuming, as it generally requires a human expert, further limiting the number of available HSI datasets for classification tasks.

These challenges greatly worsen the limitations already exhibited by DNN models (Nogueira et al., 2017), which are related to the complexity of the classifiers, such as the number of parameters required by deep models. In the following, we enumerate some of the aforementioned issues:

1. The training of DNNs is complex, since the optimization and the tuning of parameters in deep models is a non-convex and NP-complete problem (Blum and Rivest, 1989), much harder to train and without guaranteeing the convergence of the optimization process (Chen and Wang, 2014; Nguyen and Hein, 2018). Also, the increase in the number of parameters in deeper architectures often leads to multiple local minima (Bach, 2017).
2. Resulting from the large amount of parameters that must be managed in a deep model, there is a high computational burden involved, requiring computationally expensive and memory-intensive methods (Cheng et al., 2017b)
3. Also, due to the number of parameters that must be fine-tuned, supervised deep models consume great amounts of training data, and they tend to overfit when there are few training parameters (Erhan et al., 2010). In this context, the high-dimensional nature of HSI data, coupled with the limited availability of training samples, makes DNNs quite ineffective in generalizing the distribution of HSI data, requiring excessive adjustments at the training stage, while the performance on the test data is generally poor.
4. Moreover, simply stacking of layers by itself does not achieve the desirable improvement in precision results. In fact, forward propagation suffers from an important degradation of the data (He et al., 2016), while the backpropagation mechanism presents difficulties in propagating the activations and gradient signal to all layers as the network's depth increase (Srivastava et al., 2015). The gradient (which is necessary to update the model's parameters) fades slightly as it passes through each DNN layer. This degradation becomes quite severe in VDNNs, resulting in its practical disappearance or vanishing. These problems elongate the model's objective function until the model cannot properly change its weights at each iteration.
5. The "black box" nature of the training procedure is also a disadvantage, being the model's internal dynamics very hard to interpret (Benítez et al., 1997; Lipton, 2016). This may hinder the design and implementation of optimization decisions, although several efforts have been done to visualize the parameters of DNN models (Shwartz-Ziv and Tishby, 2017), and to enhance the extraction of more significant and interpretable filters.

The combination of the aforementioned challenges introduced by HSI data and the limitations of deep models force developers to carefully select and implement those models that best suit HSI data, choosing the architectures, learning strategies and improvement tricks that best fit the data while maintaining computational efficiency. In the following sections, these points will be covered in detail, providing a list of current models and methods that have been successfully applied to HSI data classification.

3. Deep neural networks: flexible and configurable models

Standard ANN models for HSI data classification exhibit a rather limited performing, usually conducting supervised learning of purely spectral features in a fully-connected architecture. On the contrary, DNNs offer a great variety of models, allowing for the inclusion of different layers, the exploitation of features in both the spectral and spatial domains, and the adoption of different learning strategies. In the following, these concepts will be briefly introduced.

3.1. Type of features

The type of features obtained from HSI data $\mathbf{X} \in \mathbb{N}^{n_1 \times n_2 \times n_{bands}}$ are one of the factors that impose several restrictions in the performance of the classifier, being crucial to the discrimination between the different classes. In particular, HSI data are characterized by their two spatial components: $n_1 \times n_2$, and by their large spectral domain, n_{band}, allowing the exploitation of both types of features (Landgrebe, 2002). Although there are also many traditional ML methods that allow for the exploitation of these two types offeatures, DNN models stand out for their versatility, adapting both their input and their internal operation to the use of such features through the implementation of different types of layers.

Focusing on traditional pixel-wise DNN classifiers, these methods exploit the ability of HSI data for detecting and uniquely characterizing the captured surface materials in certain land cover classes, learning existing relationships between the spectral signatures associated to each HSI pixel and the information that is contained in them (Chen et al., 2014b). In this sense, spectral-based DNN models learn spectral feature representations from $\mathbf{X}$, processing each pixel vector $\mathbf{x}_i \in \mathbf{X}$ in a way that is completely isolated from the rest of the pixels in the image (Romero et al., 2016), under the assumption that each $\mathbf{x}_i$ contains a perfect and pure signature of a single surface material, without any mixing of different land cover materials (Fisher, 1997). The performance and final accuracy of these classifiers is strongly related to the available training samples, usually requiring a large number of them to properly learn the parameters of the classifier (Hu et al., 2015) and to deal the spectral intraclass variability and interclass similarity –with the aim of avoiding the misclassification of the samples (traditional “salt & pepper” noise)– (Huang and Zhang, 2013).

To deal with these limitations, recent research has demonstrated the benefits of exploiting the spatial arrangement of HSI data (Jiménez et al., 2005; Zhang et al., 2012; Huang and Zhang, 2013), enhancing the classification performance of standard pixel-wise HSI classification procedures (Tarabalka et al., 2010; Mei et al., 2016) by analyzing the contextual information around each pixel $\mathbf{x}_i$ (Fauvel et al., 2008; Tarabalka et al., 2009; Bioucas-Dias et al., 2013). With advances in remote sensing technology, the spatial resolution has become gradually better, making HSI data cubes able to represent target zones/objects using finer spectral pixels and increasing the number of captured samples for each type of coverage (which intrinsically increases the intraclass variability), and improving the acquisition and observation of certain spatial patterns present in particular land cover materials. These classifiers operate under the assumption that adjacent pixels commonly belong to the same land cover category (Mura et al., 2010; Ghamisi et al., 2018), providing additional valuable information to the classification task which helps to reduce the intraclass variance and the label uncertainty. In the available literature, the contextual information given by the spatial arrangement of the HSI data cube can be employed by two kind of DNN classifiers: (i) those that only exploit the spatial features, and (ii) those that combine both spatial and spectral features to perform the final classification.

Focusing on spatial-based DNN classifiers, these models usually process some spatial information extracted from the original data cube $\mathbf{X}$, learning only spatial feature representations from the data (Chen et al., 2016). Although some spatial models may employ spatial handcrafted features as input data, such as the minimum noise fraction (MNF) (Zhang et al., 2019a) and covariance matrices (He et al., 2018), Gabor filtering (Chen et al., 2017b; Kang et al., 2018), among others, the most common and simple strategy to perform spatial HSI classification is to feed the network with some features extracted by the principal component analysis (PCA) method (Wold et al., 1987; Jolliffe, 2002; Fernandez et al., 2016), which reduces the spectral redundancy and the number of dimensions while keeping the spatial information intact (Yue et al., 2015; Haut et al., 2019a). In this context, although there is no consensus on the number of bands to be reduced, a DNN model is generally considered to be spatial when it applies PCA to the input data and its architecture allows only spatial features to be extracted (Makantasis et al., 2015; Chen et al., 2016; Haut et al., 2018c).

Although spatial-based DNN models may overcome spectral methods under some circumstances, in particular, in high spatial resolution HSI scenes with clear and distinctive spatial structures (and with spectral signatures that are not mixed) (Chen et al., 2016), the joint exploitation of both spatial and spectral features is more desirable, as it not only comprises the analysis of spectral signatures but also the associated contextual information (Paoletti et al., 2017a; Paoletti et al., 2018a). In this regard, available DNN architectures are able to process both features by including spatial information as concatenated information to the spectral vector (following the traditional ML vector vision (Chen et al., 2014b; Chen et al., 2015)), or by processing the 3-dimensional cube to maintain the original structure and contextual information (Chen et al., 2016; Paoletti et al., 2017a; Paoletti et al., 2018a; Paoletti et al., 2018c).

3.2. Type of layers

As mentioned before, the type of layer has a decisive influence on the architecture of the model, allowing for the processing of different features. Following the previous notation, DNN models divide the global mapping function $f_c(\cdot,\theta)$ into hierarchically stacked submapping functions $f^{(l)}(\cdot,\theta^{(l)})$. In this regard, each $f^{(l)}$ performs a two-step stage, composed by FE and detection, which are in turn implemented by several types of stacked layers, being $\theta^{(l)}$ their parameters.

Contextualizing the evolution of DL methods, at early days, neural models emerged within the fields of pattern recognition and signal processing, inspired by the behaviour of the biological brain and implementing a hierarchical structure where each part of the stack conforms a layer, being neurons (also perceptrons) the basic unit of each layer (Ball et al., 2017). However, with the development of image processing, traditional fully-connected structures became ineffective for the analysis of 2-dimensional and 3-dimensional data cubes (LeCun et al., 2015). To overcome this limitation, a fully-connected structure was adapted to the behaviour of those neurons that compose the biological visual cortex, characterized by a local receptive field in which they are activated or not in the presence of some specific visual *stimuli*, creating a hierarchical structure in which deeper neurons are able to respond to more abstract and higher level features. With this in mind, DNN models can implement several types of layers, where the most common ones are explained below.

3.2.1. Fully-connected layers

Also known as FC layers, they connect every neuron in the l-th layer to every neuron in the subsequent layer $l+1$, as it can be observed on the leftmost model in Fig. 1, where a traditional feed-forward multilayer perceptron (MLP) (Collobert and Bengio, 2004) is represented. These layers apply a linear transformation between the input layer data $\mathbf{X}^{(l-1)}$ and the layer parameters, weights $\mathbf{W}^{(l)}$ and biases $b^{(l)}$, adapting the original mapping function of Eq. (1) as follows:

$$\mathbf{X}^{(l)} = \mathbf{W}^{(l)}.\mathbf{X}^{(l-1)} + b^{(l)} \tag{2}$$

The main drawback of FC layers is the high number of connections, imposing a large number of parameters that must be fine-tuned. In

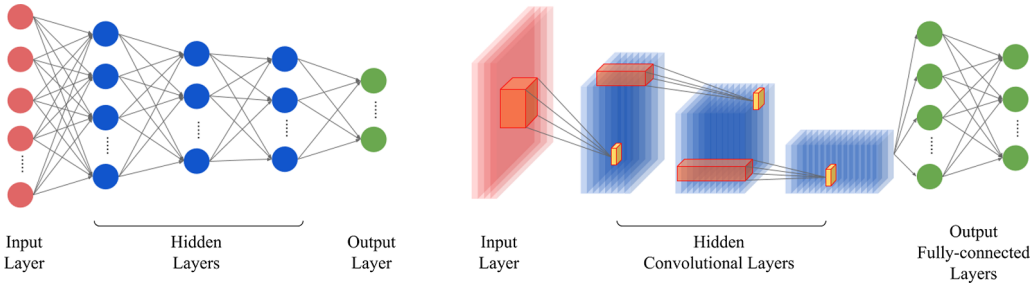


Fig. 1. Comparison between the traditional fully-connected (left) and the convolutional architecture (right) of a DNN model. The first model is represented as a conventional multilayer perceptron (MLP) with 3 hidden fully-connected (FC) layers, while the second model is represented as a convolutional neural network (CNN) with 3 hidden convolution layers too. Focusing on the last one, neurons in the CNN create 3-dimensional blocks with local connectivity over one pre-defined window of each layer input volume, known as receptive field. FC layers can be observed at the architecture tail, conforming the classifier network.

particular, the number of parameters can be calculated as the sum of all the connections between adjacent layers $n_{parameters} = \sum_{i=0}^{L-1} (n_{nodes}^{(l)} \cdot n_{nodes}^{(l+1)} + 1)$, which involves the number of weights and the bias. Also, both the input data that they need and the extracted features are limited to a vector representation of the input data, losing to some extent the potential of the spatial-contextual information (Chen and Wang, 2014).

3.2.2. Convolutional layers

As we can observe in Fig. 1, the CONV layer defines a block of neurons that operate as linear kernels (also called *filter bank*) connected and applied over small pre-defined regions from the input data (input volume hereinafter). The main idea lies on analyzing the statistical properties of the HSI cube $\mathbf{X} \in \mathbb{N}^{n_1 \times n_2 \times n_{bands}}$, which can be considered as a stationary source of spectral pixels in which data features are equally distributed into the entire $\mathbf{X}$ in relation to spatial positions (Field, 1999). This suggests that the learned features at a certain position of $\mathbf{X}$ can be successfully applied to other regions of $\mathbf{X}$, which in the end can be understood as the chance to employ the same features at all locations of the input image.

In this sense, CONV layers can be interpreted as a traditional sliding window method, where $K^{(l)}$ fixed-size filters are overlapped over the input layer data, sliding at certain intervals defined by the stride of the layer $s^{(l)}$. This can be observed in Fig. 2. In contrast with FC layers, CONV layers offer a great versatility, since the size of these chunks or windows is defined by the receptive field of the layer, indicated as $k^{(l)} \times k^{(l)} \times q^{(l)}$, where $k^{(l)}$ is applied over the two spatial axes and $q^{(l)}$ is applied over the spectral axis. This allows the CONV layer to accept 1-D, 2-D and 3-D inputs, and to extract spatial, spectral or spatial-spectral features.

$$\mathbf{X}^{(l)} = (\mathbf{W}^{(l)} * \mathbf{X}^{(l-1)} + b^{(l)})_{K^{(l)} \times k^{(l)} \times k^{(l)} \times q^{(l)}} \tag{3a}$$

$$x_{i,j,t}^{(l)z} = \sum_{\hat{i}=0}^{k^{(l)}-1} \sum_{\hat{j}=0}^{k^{(l)}-1} \sum_{\hat{t}=0}^{q^{(l)}-1} (w_{\hat{i},\hat{j},\hat{t}}^{(l)} \cdot x_{(i \cdot s^{(l)}+\hat{i}),(j \cdot s^{(l)}+\hat{j}),(t \cdot s^{(l)}+\hat{t})}^{(l-1)}) + b^{(l)} \tag{3b}$$

As Eq. (3a) indicates, the l-th CONV layer applies $K^{(l)}$ linear 3D-kernels over the input layer $\mathbf{X}^{(l-1)}$, which performs a dot product between its weights and biases, $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$, respectively, and small chunks of the input volume data. As a result, an output volume $\mathbf{X}^{(l)}$ composed by $K^{(l)}$ feature volumes is obtained. In particular, Eq. (3b) indicates the general calculation of the feature (i, j, t) for the z-th feature of the output volume, $x_{i,j,t}^{(l)z}$.

CONV layers exhibit some advantages over traditional FC layers (Guo et al., 2016; Li et al., 2017b). In particular, the local connectivity allows to learn spatial correlations among neighboring pixels, introducing some invariance to the location of the feature. Also, the sparse connectivity and the parameter sharing mechanism reduces the number of parameters that must be fined-tuned.

3.2.3. Activation layers

Usually, the data transformations applied by FC and CONV layers are considered the FE stage of the network, defining a linear operation of element-wise matrix multiplication and addition over the data. In this sense, those DNN models without activation layers (or with linear activation ones) are essentially working as linear regressors. A non-linear activation layer must be implemented behind FC and CONV layers in order to learn non-linear representations of the data structure. In fact, the activation layer is considered as the detector stage of DNN models (Goodfellow et al., 2016), and is implemented by a non-linear, element-wise activation function which allows to model a response variable (i.e., a feature score) that varies non-linearly with the output volume of the previous FC/CONV layer, giving as a result an output volume containing the activations of each neuron of the previous layer, $\mathbf{X}^{(l)} = \mathcal{H}(\mathbf{X}^{(l-1)})$. In this regard, $\mathcal{H}(\cdot)$ can be implemented by several activation functions, depending on the desired properties. Fig. 3 gives the graphical visualization of some widely used functions. For instance,

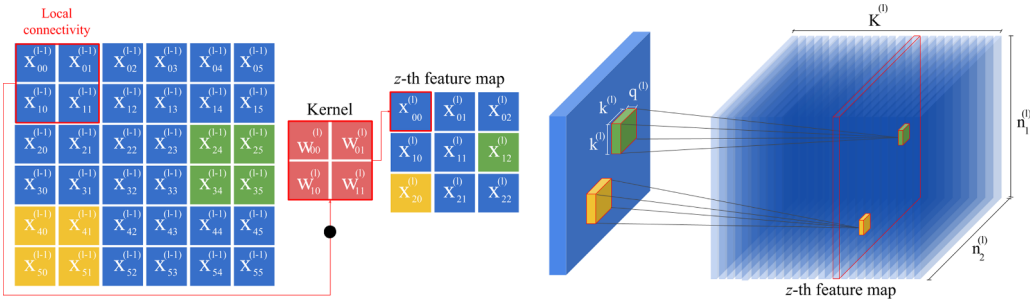


Fig. 2. Graphical visualization of the CONV layer from a 2D point of view (left) and 3D point of view (right). On the left we can observe how the 2D kernel is applied over spatial regions of the input volume $\mathbf{X}^{(l-1)}$ with a stride $s^{(l)} = 2$ (the dark circle symbolizes the dot product between the window from the original data and the kernel). On the right we can observe how the z-th kernel of size $k^{(l)} \times k^{(l)} \times q^{(l)}$ produces, for each region to which it is applied, a scalar value (represented as a smaller rectangle) which is allocated into the z-th feature map, composing an output volume $\mathbf{X}^{(l)}$ of $K^{(l)}$ feature maps.

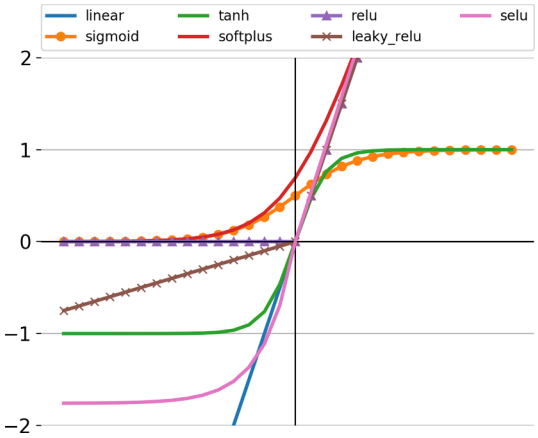


Fig. 3. Graphical visualization of different activation functions that can be implemented in a DNN model, including the linear function $\mathcal{H}(x) = x$, the *leaky* ReLU, and SeLU (Section 5.3.3 contains further details).

the *sigmoid* $\mathcal{H}(x) = \frac{1}{1+e^{-x}}$ presents a smooth and continuously differentiable function, whose values range from 0 to 1 (not inflating the neural activation values). However, as it only produces positive values in the 0–1 range, it becomes hard and slow to optimize. The *tanh function* $\mathcal{H}(x) = \frac{e^{x}-e^{-x}}{e+e^{-x}}$ is very similar to the sigmoid, being less smooth and symmetric over the origin, as its values range from -1 to 1. This makes its gradient steeper than that of the sigmoid.

Although these standard activations can operate properly with shallow architectures, the smallest derivative terms tend to zero when the model's architecture is deep enough, leading to the vanishing gradient problem. The *rectified linear activation function* (ReLU) (Nair and Hinton, 2010) tries to overcome previous limitations by applying a $\max(\cdot)$ function between 0 and the input data x, setting the gradient to 0 if the data are equal or smaller than 0, and to x otherwise, i.e. $\mathcal{H}(x) = \max(0, x)$, with an output range of $[0, +\infty)$ (it is unbounded on the positive side). This alleviates the vanishing gradient problem, as the derivative of the positive x is always 1. Moreover, ReLU conducts a sparsity activation function where not all the neurons are activated at the same time, being more computationally efficient than the sigmoid, for instance. However, if the gradient is set to 0, the influence of the affected neurons is eliminated, so they cannot contribute to improving the learning process (Pedamonti, 2018), leading to the *dying ReLU* problem.

Other interesting functions are the *softplus* (Dugas et al., 2001) and *softmax* activation functions, with equations $\mathcal{H}(x) = \ln(1 + e^{x})$ and $\mathcal{H}(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$, respectively. The first one produces values in the range $[0, +\infty)$, i.e. it is similar to a smoothed ReLU being differentiable into 0 and where its derivative is the sigmoid function, which makes it computationally slower during the backward step. The second one is inspired by the sigmoid, squeezing the input layer data between 0 and 1 and dividing the obtained outputs by the sum of them. In this sense, the softmax function works as a winner-take-all function that gives the probability of the input data belonging to a particular class, and it is usually employed as the final layer of a DNN model.

Despite the wide range of activation functions available in the current DL literature (Agostinelli et al., 2014; Sonoda and Murata, 2017; Ramachandran et al., 2017), the vast majority of DNN models for the analysis of HSI remote sensing data employ ReLU and softmax as the principal activation functions, with few exceptions (Mei et al., 2016; Paoletti et al., 2018).

3.2.4. Down-sampling layers

Also known as pooling or POOL layers, they are inspired by the spatial processing of CONV layers. Particularly, POOL layers perform a non-linear sub-sampling strategy with the aim of: (i) reducing the spatial dimensions of the extracted feature maps, sumarizing them into a reduced volume, (ii) contributing to the data with certain invariance to small transformations, and (iii) reducing the computation time and the complexity in terms of both, data size/dimensionality and network parameters (Boureau et al., 2010). The POOL layer implements a sample-based discretization process (see Fig. 4), applying some numerical operation over a square window defined by the spatial receptive field $k^{(l)} \times k^{(l)}$ of the layer. The most usual operations are the average-pooling, the sum-pooling or the max-pooling (Scherer et al., 2010), although it should be noted that several alternative methods have been also implemented, such as stochastic pooling (Zeiler and Fergus, 2013), mixed pooling (Yu et al., 2014) or wavelet pooling (Williams and Li, 2018). Also, several works have investigated the replacement of pooling layers by CONV layers with increased stride (Springenberg et al., 2014).

3.3. Learning strategies

In addition to the type of features and layers used, DNN models also allow the implementation of different learning strategies. Following the previous notation, the classification function $f_c(\cdot,\theta)$ can be understood as a particular DNN model. In this sense, the performance of f_c will depend on certain parameters θ that must be correctly fine-tuned. Moreover, depending on how this parameter adjustment is carried out, two main types of learning can be distinguished: *unsupervised* and *supervised* learning

3.3.1. Unsupervised learning

Unsupervised learning performs the classification without *a priori* knowledge about the given data, optimizing parameters θ by the inherent similarities present in the data structure (Xiaoli Jiao, 2007; Romero et al., 2016; Hassanzadeh et al., 2018). In this context, an unsupervised DNN performs a *greedy layer-wise unsupervised pre-training* (Bengio et al., 2013), where each layer performs hierarchical unsupervised FE for inferring the underlying structure of the data, being

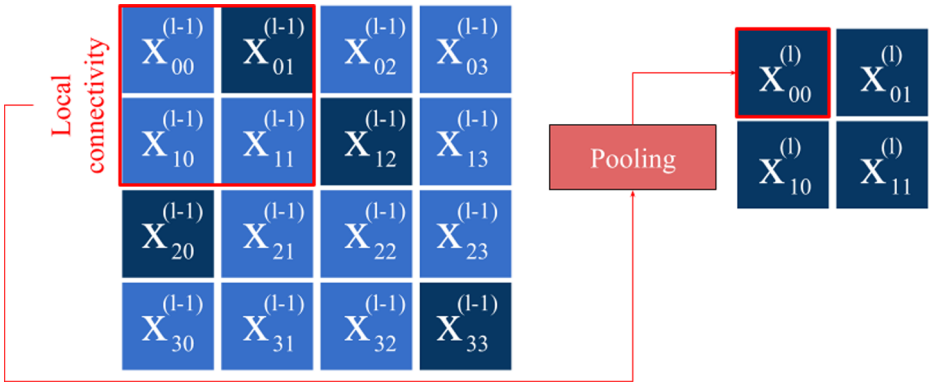


Fig. 4. Graphical visualization of the POOL layer from a 2D point of view. Dark cells indicate the selected value from the pooling operation (for instance, if the max pooling has been implemented, dark cells would represent the higher value of each region from the volume), although the final value also can be obtained as the average or sum value of the entire region. In fact, the pooling layer can be interpreted as a kernel of size $k^{(l)} \times k^{(l)}$.

combined at the end to initialize another deep supervised or generative model (Bengio et al., 2012) that will carry out the final regression or classification task. In fact, unsupervised DNN models are usually employed for clustering (Xie et al., 2016; Tian et al., 2017; Shaham et al., 2018; Min et al., 2018), anomaly detection (Penttilä, 2017; Ma et al., 2018a) and data encoding (Li et al., 2014; Paul and Kumar, 2018; Kang et al., 2018). In particular, there is a wide range of works about unsupervised DL methods for HSI data pre-processing, being widely used to perform dimensionality reduction (DR) (Lin et al., 2013b).

3.3.2. Supervised learning

In contrast to unsupervised learning, supervised learning needs to learn those parameters θ that model the relationship between $\mathbf{x}_i$ and $\mathbf{y}_i$ by performing an inference procedure based on previous knowledge (Sabale and Jadhav, 2015; Qiu et al., 2017). In this way, it is needed to split the original scene $\mathbf{X}$ into those training samples with known identity $\mathcal{D}_{train} = \{\mathbf{x}_i, y_i\}_{i=1}^{n_{labeled}}$ that will be used during the training step to categorize the rest of unlabeled pixels, which will be employed during the inference $\mathcal{D}_{test} = \{\mathbf{x}_i\}_{i=1}^{n_{unlabeled}}$ (Sabalel and Jadhav, 2014). Usually, supervised DNNs models are able to achieve better performance than their unsupervised counterparts, being the most widely used (Chen and Wang, 2014; Sabale and Jadhav, 2015; Qiu et al., 2017; Paoletti et al., 2017a; Paoletti et al., 2018a). However, this learning also imposes a severe training constraint, whereby DNN models need to consume large amounts of labelled data during the training to correctly fine-tune the model parameters (Makantasis et al., 2015).

4. When HSI data meets DL: main classification models

To date, in addition to the traditional MLP (Roodposhti et al., 2019), four DNN models have become the mainstream DL architectures for the analysis of HSI data: autoencoders (AEs), deep belief networks (DBNs), recurrent neural networks (RNNs) and CNNs. In the following we review each model, pointing the most relevant works in the HSI literature, and then paying more attention to state-of-the-art models such CNNs.

4.1. Autoencoders (AEs)

When dealing with HSI data classification tasks, the extraction of accurate features becomes a critical preprocessing step to model the internal structures and relationships of the data, helping to reduce the Hughes effect and the curse of dimensionality. In this sense, auto-associative neural networks (AANNs), also known as autoencoders (AEs) (Hinton and Zemel, 1993; Bishop, 1995; Chen et al., 2014b; Karhunen et al., 2015; Zhang et al., 2016c; Plaut, 2018) have been widely used as deep models to perform unsupervised coding from HSI data. Regarding its operational mode, the AE model does not carry out classification tasks, but reconstructs the input data by reducing $\min\|\mathbf{X} - \mathbf{X}'\|_2$ (the distance between the obtained representation $\mathbf{X}'$ and the original data $\mathbf{X}$). In fact, the main particularly of these networks lies in their ability to project the original input samples into a new space, generating compressed, extended or even equally-dimensioned outputs, with the least possible amount of distortion. This projection is performed by a traditional architecture implemented by encoder and decoder nets, both linked by a bottleneck layer that represents the latent space (Baldi and Hornik, 1989; Hinton and Zemel, 1993), as it can be observed in Fig. 5.

HSI-AEs emerged as typically pixel-wise methods, being usually exploited to carry out dimensionality reduction (DR) and high-level spectral FE due to the existing correlation between adjacent bands (Ahmad et al., 2017). In this regard, the spectral pixel $\mathbf{x}_i \in \mathbb{N}^{n_{bands}}$ is taken as input of the encoder, representing it in a new space $\mathbb{R}^{n_{new}}$ by applying a hierarchical set of $L_{encoder}$ recognition weights or encoder components, as Eq. (4a) illustrates. Then, the obtained code vector or code dictionary $\mathbf{c}_i \in \mathbb{R}^{n_{new}}$ is sent as an input to the decoder, which applies a set of $L_{decoder}$ generative weights over the code vector to recover and/or obtain an approximate reconstruction of the original input vector, $\mathbf{x}_i'$, as Eq. (4b) indicates.

$$\mathbf{c}_i \leftarrow \text{For } \; l \text{ in } L_{encoder}: \quad \mathbf{x}_i^{(l+1)} = \mathcal{H}(\mathbf{W}^{(l)} \cdot \mathbf{x}_i^{(l)} + b^{(l)}) \tag{4a}$$

$$\mathbf{x}_i' \leftarrow \text{For } \; ll \quad \text{in } \; L_{decoder}: \quad \mathbf{c}_i^{(ll+1)} = \mathcal{H}(\mathbf{W}^{(ll)} \cdot \mathbf{c}_i^{(ll)} + b^{(ll)}) \tag{4b}$$

Several AE models for HSI data analysis have been presented in the literature. Focusing on spectral-based ones, Zhu et al. (2017a) propose an unsupervised tied AE (TAE) for spectral FE, based on the maximum noise fraction (MNF) (Green et al., 1988; Iyer et al., 2017) as pre-processing DR step, and fine-tuning with classification via softmax. Following a simple architecture, Hassanzadeh et al. (2017) combine the multi-manifold spectral clustering (MMSC) (Wang et al., 2010) with the unsupervised contractive AE (CAE) (Rifai et al., 2011) to enhance the HSI data classification by reinforcing the model's learning through a regularizer term, being less sensitive to small variations in the training samples. A pixel-wise stacked AE (SAE) is proposed by Okan et al. (2014), which implements a two-step training strategy with unsupervised representation learning and supervised fine-tuning, before the final supervised classification, performed by a logistic regression layer. Furthermore, Wang et al. (2016) implement a stacked denoising AE (SDAE), which stochastically corrupts the inputs in order to overcome the *identity-function risk* present in deep AEs. Also, in order to reduce the computational complexity of SAEs, Zabalza et al. (2016) propose a segmented SAE (S-SAE) to comprise original features into smaller data segments, being separately processed by smaller and independent SAEs.

Also, recent works combine AEs with spectral-spatial feature extraction methods. For instance, Chen et al. (2014b) present three different AEs and SAEs to generate shallow and deep or high-level features using spectral, spatial and spectral-spatial information, using a logistic regression method to perform the final classification, while Lin et al. (2013b) perform a comparison between spectral and spectral-spatial AEs with shallow and deep architectures. In both cases, the spatial information is obtained via PCA reduction, obtaining n_{new} components and flattening the $d \times d \times n_{new}$ cube that surrounds each pixel into a vector. Mughees et al. (2016) also develop a SAE to perform spectral processing, while spatial analysis is performed by an adaptive boundary adjustment-based segmentation method. As a result, the spectral-based classification map and the spatial-based single band segmented map are combined by a majority voting based method. Wang et al. (2017b) apply guided filtering (He et al., 2013) to exploit the spatial information, flattening it to combine it with spectral information in a multilayer fine-tuning SAE (FSAE). Li et al. (2015a) implement a SAE, which is pre-trained in unsupervised fashion over 3D Gabor features extracted from the HSI data cube, with an MLP performing the final classification. Ma et al. (2016b) combine the FE performed by the SAE with a relative distance prior in the fine-tuning process, in order to enhance the model when the number of available labeled samples is not enough. Also, Ma et al. (2016a) introduce a spatial updated deep auto-encoder (SDAE) to improve the extraction of spectral-spatial information by adding a regularization term in the energy function, and updating the features by integrating contextual information. Paul and Kumar (2018) propose a segmented stacked autoencoder (S-SAE) for spectral-spatial HSI data classification as an improvement of the SAE, reducing its complexity and computational times through the use of mutual information (MI), to perform spectral segmentation, and morphological profiles (MPs) to assimilate the spatial information contained in the HSI cube. Tao et al. (2015) develop two stacked sparse AEs (SSAEs) to extract overcomplete sparse spectral and spatial features, which are stacked and embedded into a linear SVM for classification purposes. Wan et al. (2017) also propose a SSAE to process different types of features, such as spectral-spatial, multifractal and other higher-order statistical ones, while a RF is employed for classification. Zhao et al. (2017a) exploit again the SSAE with RF to extract and classify more abstract and deep-seated

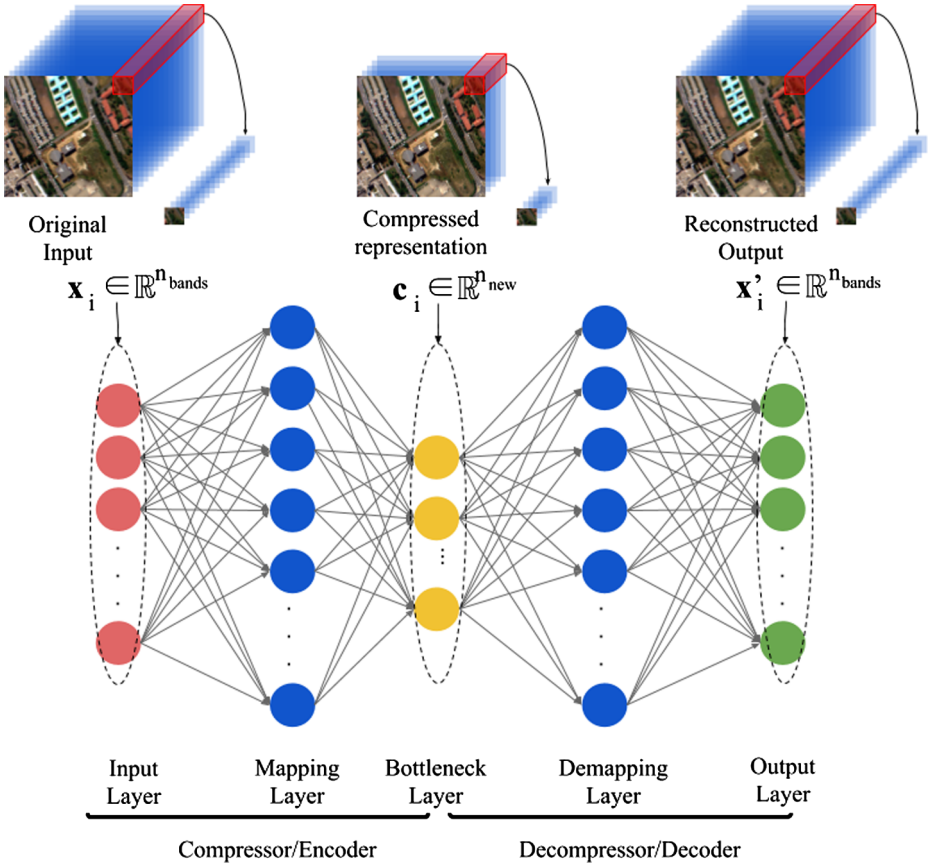


Fig. 5. Traditional representation of a tied autoencoder, composed by two main parts: an encoder and a decoder, linked by a bottleneck layer.

features from spectral, spatial and spectral-spatial sets. In contrast, Xing et al. (2016) develop a SDAE to extract robust spectral features from HSI data, using logistic regression to perform the supervised fine-tuning and classification. Liu et al. (2015) also employ a SDAE to learn spectral feature representations from the input data, while a superpixel technique is employed to generate the spatial constraints for refining the spectral classification results. Recently, Zhou et al. (2019a) have proposed a two-stage AE, called compact and discriminative SAE (CDSAE), where the first one performs the training of a discriminative SAE (DSAE, where each layer performs a local Fisher discriminant regularization) to learn a feature mapping by minimizing the reconstruction error, and the second one performs the classification of the data, updating the DSAE's parameters. Also, the extraction of spectral features using AEs has been combined with neural models such as CNNs (to extract spatial information), as Hao et al. (2018) discussed.

Although AE structures have demonstrated to be a powerful tool, their performance is often hampered by the large number of parameters that must be trained, learned and updated, which requires a large number of samples to perform the fine-tuning process, a demand that cannot be always satisfied. Although several new techniques have been adopted to avoid this problem, such as the use of active learning (AL) (Li, 2015), additional enhancements are needed. Furthermore, the spatial processing step that AEs usually perform on the data implies the use of DR methods followed by a flattening of the data into a vector, neglecting the rich spectral-spatial structural information that HSI data cubes contain (Chen et al., 2014a; Tuia et al., 2015).

4.2. Deep belief networks (DBNs)

DBNs combine probability and graph theory to implement a generative probabilistic graphical model (PGM) with the structure of a directed acyclic graph (DAG) (Ball et al., 2017). In the literature, several works address the implementation of DBNs as a stack of unsupervised networks, such as restricted Boltzmann machines (RBMs) (Smolensky, 1986; Larochelle and Bengio, 2008; Tan et al., 2019) with a greedy learning algorithm as optimizer (Hinton et al., 2006; Hinton and Salakhutdinov, 2006).

In HSI data analysis, DBNs have been employed as a variant of the AE model with greedy layer-wise training to perform FE. In this sense, Li et al. (2014) implement a DBN for feature extraction and classification, stacking spectral-spatial characteristics and using logistic regression for classification. Also, Chen et al. (2015) introduce three DBNs to extract spectral, spatial and spectral-spatial high-level features from HSI data in hierarchical fashion, and performing the final classification task by means of logistic regression. There are also several efforts aimed at improving the performance of this kind of DNN for HSI classification purposes, for instance Le et al. (2015) review the hyper-parameters used by the spectral and spectral-spatial DBNs of Chen et al. (2015), while Zhong et al. (2017a) present a diversified DBN for HSI data classification, which regularizes the pre-training and fine-tuning procedures by a diversity-promoting prior over latent factors to avoid the co-adaptation of the latter. Guofeng et al. (2017) improve the standard training process of DBNs in order to avoid the effect of a gradient disappearance, using PCA and kernel PCA (KPCA). Inspired by DBNs, Zhou et al. (2017) developed a group belief network (GBN), which considers the characteristics of grouped spatial-spectral features from HSI data by modifying the bottom layer of each RBM that composes the model architecture.

Although DBNs are very promising DL methods for HSI data classification, as they often provide good results (and improve their performance with the incorporation of spatial information), they suffer from the same limitation as SAEs: these neural models are designed for processing 1D-signals, so the rich spatial information contained in HSI data cubes must be vectorized to be processed together with the

spectral one, or even separately processed by other techniques in order to be properly exploited. In the end, this kind of spectral-spatial processing cannot fully incorporate the spatial-contextual information present in HSI data cubes.

4.3. Recurrent neural networks (RNNs)

The architecture of RNN models (Williams and Zipser, 1989) is characterized by loops in the connections, where node-activations at each step depend on those of the previous step. This internal structure (similar to a directed graph) makes the RNN an ideal model for learning temporal sequences, exhibiting a dynamic temporal behavior for a given data sequence, with an internal state or memory that allows for the association between the current input data and the previous ones at each step (i.e. remembering the context). This fact enables RNNs as a powerful tool for predicting future events depending on the previously remembered ones, being particularly interesting for remote sensing land-cover analysis, which exhibits many changes in their reflective characteristics over time, hampering the classification task (Rußwurm and Körner, 2017).

RNNs can be categorized into three main groups: vanilla RNNs, long short term memory (LSTM) (Hochreiter and Schmidhuber, 1997) and gated recurrent unit (GRU) (Cho et al., 2014) architectures. The vanilla RNN was the first recurrent model introduced as a DL framework, and its operation is quite intuitive. Given an input data sample $\mathbf{x}_t \in \mathbb{R}^n$ captured at time t, the vanilla RNN computes its corresponding output $\mathbf{y}_t$ as a hidden state at time t, $\mathbf{y}_t = \mathbf{h}_t$, which represents the current memory of the model, as Eq. (5) indicates:

$$\mathbf{h}_t = \begin{cases} 0 & \text{if } t = 0 \\ \mathcal{H}(\mathbf{W}_h \cdot \mathbf{x}_t + \mathbf{U}_h \cdot \mathbf{h}_{t-1} + b_h) & \text{if } t \neq 0 \end{cases} \tag{5}$$

where $\mathcal{H}(\cdot)$ is a non-linear activation function (for instance, the sigmoid), b_h is the bias, $\mathbf{W}_h$, is the weight matrix of the input, and $\mathbf{U}_h$ is the weight matrix of the recurrent connections.

Although vanilla is the easiest RNN model to implement, its simplicity leads to a degradation of information when high dimensional input data are processed. In this sense, the LSTM offers advantages when dealing with the deficiencies of the original RNN by developing a recurrent unit composed by a cell, which remembers values at arbitrary time intervals, and three gates (input, output and forget gates), intended to regulate the flow of information in and out of the cell. A schematic overview is presented in Fig. 6. In this case the model stores, for each input at time t, two states: the original hidden state, $\mathbf{h}_t$, and the cell state, $\mathbf{c}_t$, which removes or adds information to the cell, depending on the gates. In particular, the input gate $\mathbf{i}_t$ determines whether or not a new input is allowed to go inside the cell, the forget gate $\mathbf{f}_t$ deletes the irrelevant or unimportant information, and the output gate $\mathbf{o}_t$ allows the information to affect the network's output at time t. This mechanism allows the LSTM unit to learn which information is important along time, as Eq. (6) indicates:

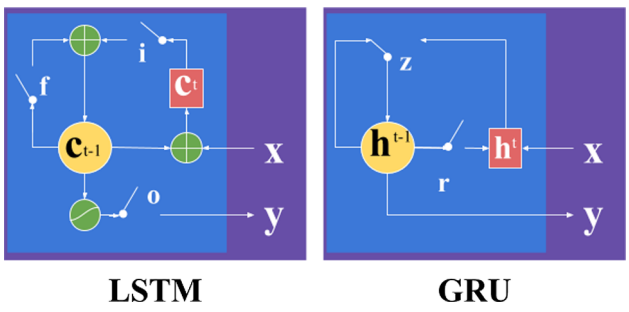


Fig. 6. Architecture of RNN models: comparison between the internal architecture of a LSTM recurrent unit and a GRU one.

$$\begin{aligned} \mathbf{i}_t &= \mathcal{H}(\mathbf{W}_i \cdot \mathbf{x}_t + \mathbf{U}_i \cdot \mathbf{h}_{t-1} + b_i) \\ \mathbf{f}_t &= \mathcal{H}(\mathbf{W}_f \cdot \mathbf{x}_t + \mathbf{U}_f \cdot \mathbf{h}_{t-1} + b_f) \\ \mathbf{o}_t &= \mathcal{H}(\mathbf{W}_o \cdot \mathbf{x}_t + \mathbf{U}_o \cdot \mathbf{h}_{t-1} + b_o) \\ \mathbf{h}_t &= \begin{cases} 0 & \text{if } t = 0 \\ \mathbf{o}_t \circ \mathcal{H}(\mathbf{c}_t) & \text{if } t \neq 0 \end{cases} \\ \mathbf{c}_t &= \begin{cases} 0 & \text{if } t = 0 \\ \mathbf{f}_t \circ \mathbf{c}_{t-1} + \mathbf{i}_t \circ \mathcal{H}(\mathbf{W}_c \cdot \mathbf{x}_t + \mathbf{U}_c \cdot \mathbf{h}_{t-1} + b_c) & \text{if } t \neq 0 \end{cases} \end{aligned} \tag{6}$$

where $\mathbf{W}_*$, $\mathbf{U}_*$ and b_* are the weight matrices and biases for the different gates or the cell (depending on $*$).

Finally, the GRU unit is a LSTM variant (see Fig. 6) in which the input and forget gates are changed by update ($\mathbf{z}_t$) and reset ($\mathbf{r}_t$) gates, removing the output gate (which implies less parameters):

$$\begin{aligned} \mathbf{z}_t &= \mathcal{H}(\mathbf{W}_z \cdot \mathbf{x}_t + \mathbf{U}_z \cdot \mathbf{h}_{t-1} + b_z) \\ \mathbf{r}_t &= \mathcal{H}(\mathbf{W}_r \cdot \mathbf{x}_t + \mathbf{U}_r \cdot \mathbf{h}_{t-1} + b_r) \\ \mathbf{h}'_t &= \tanh(\mathbf{W}_h \cdot \mathbf{x}_t + \mathbf{r}_t \circ \mathbf{U}_h \cdot \mathbf{h}_{t-1} + b_h) \\ \mathbf{h}_t &= \begin{cases} 0 & \text{if } t = 0 \\ \mathbf{z}_t \circ \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \circ \mathbf{h}'_t & \text{if } t \neq 0 \end{cases} \end{aligned} \tag{7}$$

As any traditional pixel-based approach, the RNN exploits each HSI pixel in band-to-band fashion, performing a similarity check between temporary data and spectral bands, using a many-to-one scheme such as the LSTM and GRU models for HSI data processing presented by Mou et al. (2017). Also, Guo et al. (2018) propose a LSTM model with a guided filter, taking into account three principal components extracted by PCA, and Zhou et al. (2018a) combining spectral LSTM-classification with PCA extracted spatial LSTM-classification via decision fusion. Lyu et al. (2016) develop the REFEREE change rule for a LSTM-based model in order to enhance the efficiency and performance when dealing with change detection in multispectral and HSI data. Zhang et al. (2018b) introduce the LSS-RNN, a RNN model with a local spatial sequential method (LSS) that includes a low-level FE step, implemented using Gabor filtering and differential morphological profiles (DMPs) (Huang and Zhang, 2013), whose corresponding features are stacked together and passed through the LSS to obtain higher-level features, which finally feed the RNN. Furthermore, Sharma et al. (2018) enhances the pixel-based RNN by implementing a patch-based RNN (PB-RNN) with LSTM units, which is able to process the multi-spectral, multi-temporal and spatial information contained into the dataset.

Other interesting RNN models take advance of the flexibility offered by CONV layers, including some stages of FE and detection with CONV after applying the recurrent unit. For instance, Venkatesan and Prabu (2019) employ a RNN to classify the features obtained by a spectral CNN model (developing a CNN1D), while Luo (2018) proposed a shortened spatial-spectral RNN with Parallel-GRU (St-SS-pGRU) with the aim of improving performance, increasing efficiency and simplifying the training procedure of standard band-by-band GRU models. Zhou et al. (2018b) first perform a spatial FE with CNNs, and then send the obtained features to a fusion network based on GRUs. Mou et al. (2019) implement a multi-spectral-temporal-spatial model for change detection by adopting an end-to-end network with several CONV layers (at the beginning of the architecture) in order to extract spectral-spatial features in a natural and structured way, enhancing the data representation before applying the LSTM unit and a FC layer to perform the final classification. Moreover, Shi and Pun (2018) combine the feature extraction performed by the spectral-spatial CNN model with a multi-scale hierarchical recurrent neural network (MHRNNs) that captures the spatial relations of local spectral-spatial features at different scales. Also, several convolutional RNNs (CRNNs) (Zuo et al., 2015) have been implemented for HSI classification. For instance, Wu and Prasad (2017) present a 1D-CRNN, where several 1D-convolutional layers are used to perform spectral FE, sending the obtained features to the recurrent layers, and finally integrating spatial constraints by adding linear opinion pools (LOP) (Benediktsson and Sveinsson, 2003) at the end of the flowchart in order to improve classification

performance. A similar model is used by Wu and Prasad (2018), where a 1D-CRNN is trained in a semi-supervised way with labeled and unlabeled data using pseudo labels. Yang et al. (2018) introduce the 2D-CRNN and 3D-CRNN for HSI data classification, performing a direct comparison with their CNN counterparts and demonstrating the superiority of the proposed RNN models. Finally, Liu et al. (2017d) introduce a bidirectional-convolutional LSTM (Bi-CLSTM) to learn spectral-spatial features from HSI data, while Seydgar et al. (2019) integrate a CNN model with 3D-kernels and the CLSTM network to extract low-dimensional and shallow spectral-spatial features that are recurrently analyzed, focusing on the spatial information but also considering the spectral one.

4.4. Convolutional neural networks (CNNs)

In contrast with the previous models, in which the FC layer is the basis of their architectures, in the CNN model (Lecun et al., 1998) the CONV layer is the basic structural unit, inspired by the natural vision process to perform FE (LeCun et al., 2015; Goodfellow et al., 2016). In this sense, CNN models elegantly integrate spectral features with spatial-contextual information from HSI data in a more efficient way as compared to previous DNN models. The large flexibility that this model provides regarding the dimensionality of the operational layers, their depth and breadth, and its ability to make strong assumptions about the input images (Krizhevsky et al., 2012), have turned the CNN into one of the most successful and popular DNN models, being the current state-of-the-art in DL (Gu et al., 2018) and an extremely popular tool for HSI data classification.

The architecture of a CNN is composed by two well-differentiated parts that can be interpreted as two networks. These coupled networks are trained together as an end-to-end model to optimize all the weights in the CNN: (i) the FE-net, composed by a hierarchical stack of feature extraction and detection stages that learns high-level representations of the inputs, and (ii) the classifier, composed by a stack of FC layers that performs the final classification task, computing the membership of each input sample to a certain class (Ball et al., 2017).

Focusing on the FE-net, it is composed by several hierarchically stacked extraction and detection stages, where the l-th stage defines the l-th submapping function $f^{(l)}$. Usually, these submapping functions are composed by CONV, activation or ReLU, and POOL layers (Murugan, 2017). In this way, the CNN model is able to reveal the features that are shared across the data domain via localized kernels, extracting the local stationarity properties of $\mathbf{X}$ (Defferrard et al., 2016). In fact, the feature extraction performed by the CNN is very similar to the ones adopted by other DNN models, i.e. the first stages are able to detect *recognizable* features, while the last stages combine all the features detected by the previous layers, detecting more *abstract* features. However, the flexibility when designing kernels allows for a more efficient and natural extraction of spectral, spatial and spectral-spatial features, as Fig. 7 shows, while the locally-connected nature of the convolutional kernels, coupled with the parameter-sharing across layers, permits to alleviate the number of parameters that must be fine-tuned by the model, making the computations more efficient as compared with traditional FC architectures. Focusing on the classifier net, it performs the final classification taking into account the information obtained by the FE-net. Usually, this part is implemented by several stages composed by FC and ReLU layers, placing a softmax on the last FC layer. The resulting output can be interpreted as the probability that each input data pattern belongs to a certain class, where the optimization function can be defined as the difference between all the desired outputs $\mathbf{y}_i$ (for each input data sample $\mathbf{x}_i$) and the obtained ones, $\mathbf{y}'_i$, which can be calculated as the cross-entropy of the data:

$$\phi_c = -\sum_i \mathbf{y}'_i - \log(\mathbf{y}_i) \tag{8}$$

Moreover, the classifier net can be implemented by a standard MLP model, or by other classifiers such as SVM (Paoletti et al., 2017b) or logistic regression (Zheng et al., 2017). Also, the classifier can be disregarded, using the first part, i.e. the FE-net, for other purposes such as unsupervised FE (Romero et al., 2016). In the current literature, three kinds of CNN models can be found for HSI data classification, depending on whether they perform spectral, spatial, or spectral-spatial feature analysis. In the following, we review some available works in each category.

4.4.1. Spectral CNN models for HSI data analysis

Regarding spectral models (top of Fig. 7), they consider the spectral pixels $\mathbf{x}_i \in \mathbb{N}^{n_{channels}}$ as the input data, where $n_{channels}$ can either be the number of original bands n_{bands} or a reasonable number of spectral channels n_{new}, extracted using PCA or other DR methods, to which 1D-kernels are applied on each CONV layer, $K^{(l)} \times q^{(l)}$, obtaining as a result an output $\mathbf{X}^{(l)}$ composed by $K^{(l)}$ feature vectors.

Hu et al. (2015) and Salman and Yüksel (2016) present a deep CNN with five 1-D layers that receive as input data the pixel vectors, classifying HSI data cubes only in the spectral domain, while Charmisha et al. (2018) present a CNN1D architecture called vectorized CNN (VCNN) to perform DR and classification of HSI data based on the topology of Hu et al. (2015). Li et al. (2017a) propose a CNN1D model for exploring spectral information correlated between pixels, extracting pixel pair features (PPFs) from the original data, being the input a combination of the center pixel and each of its surrounding neighbors (exploiting the similarity between pixels). Similarly, Du and Li (2018) develop subtraction PPFs, where the CNN1D model's input is the spectral difference between the central pixel and its adjacent pixels, performing HSI target detection. Mei et al. (2016) train the model by considering the spectrum of the pixel, the spectral mean of neighboring pixels, and the mean and standard deviation per spectral band of the neighboring pixels, introducing several improvements into the CNN1D architecture, such as batch normalization layers (Xu et al., 2015b), a dropout process (Krizhevsky et al., 2012) and a new nonlinear activation function known as Parametric ReLU (PReLU) (He et al., 2015). Acquarelli et al. (2018) develop seven shallow CNN1D models with spectral-locality-aware regularization (R), smoothing-based data augmentation (S) and label-based data augmentation (L), to include some kind of spatial information into the network, creating seven combinations (CNN-R, CNN-S, CNN-L, CNN-RS, CNN-RL, CNN-SL and CNN-RSL), although the spectral pixels are processed independently, i.e. one by one. Finally, Ghamisi et al. (2017a) and Chen et al. (2016) present standard CNN1D models for spectral processing.

In addition to 1-D architectures, the CNN2D architecture can be adapted to work only with spectral information. For instance, Jia et al. (2016) take into account only the pixel spectral array $\mathbf{x}_i$, which is folded into a map matrix and sent to the CNN2D as input.

4.4.2. Spatial CNN for HSI data analysis

Regarding spatial models, they only consider spatial information obtained from the HSI data cube. In this sense, it is usual to employ CNN2D architectures to process the spatial information, where each CONV layer applies $K^{(l)} \times k^{(l)} \times k^{(l)}$ kernels over the input data, obtaining as a result $K^{(l)}$ feature maps.

The spatial information can be extracted from the original HSI data cube by reducing the spectral dimension by employing some DR-method, such as PCA, and cropping spatial patches of $d \times d$ pixel-centered neighbors. For instance Chen et al. (2016) and Haut et al. (2019a) train a CNN2D with one principal component (PC), while Liang and Li (2016) employ three PCs to train the CNN2D and post-process the extracted spatial-features with sparse coding (SC) (Charles et al., 2011; Song et al., 2014) to create a sparse dictionary of more representative spatial features for classification. Xu et al. (2018) propose the random patches network (RPNet) as a CNN2D model where input data is whitened by PCA, taking into account only three PCs. Also, Zheng et al. (2017) perform an end-to-end classification with a CNN2D that receives

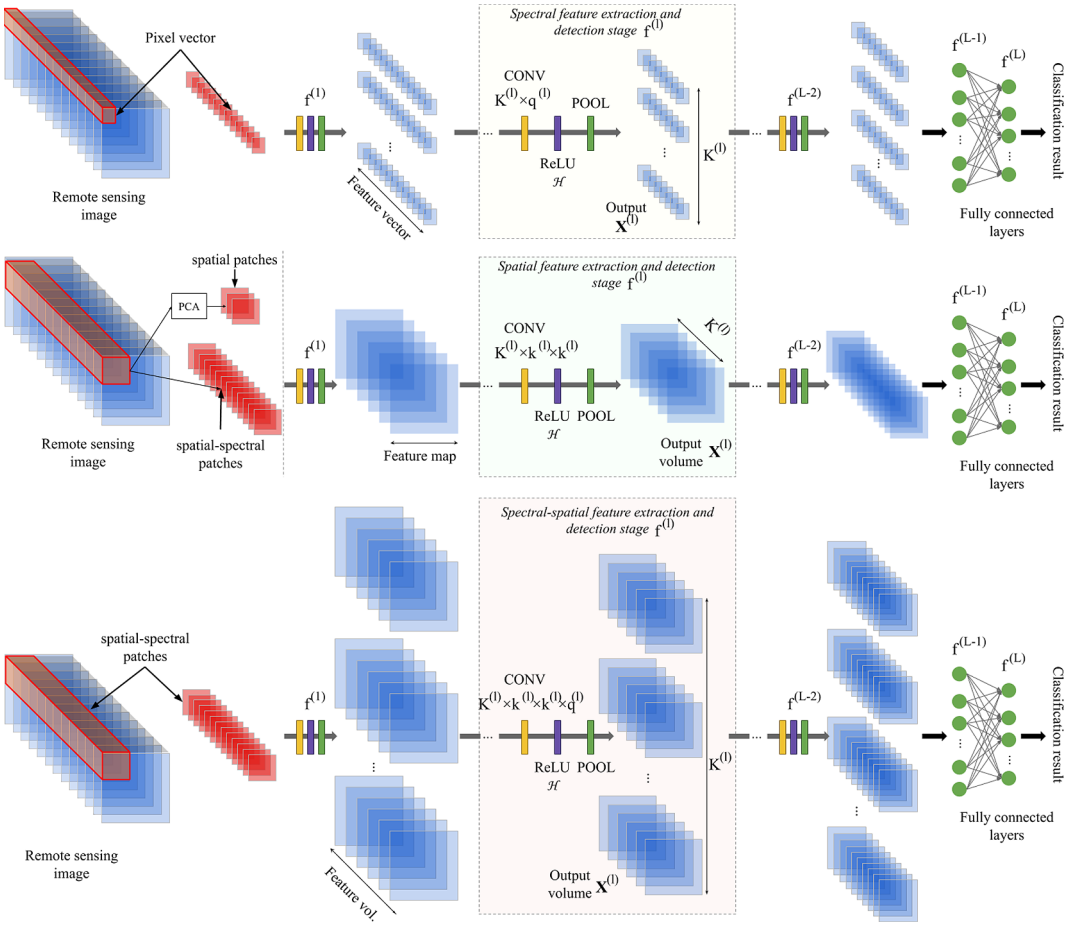


Fig. 7. Traditional architectures of spectral, spatial and spectral-spatial convolutional models employed by CNN1D, CNN2D and CNN3D architectures (top to bottom). The CNN1D architecture is commonly employed for spectral analysis, applying a hierarchical stack of L FE and detection stages, where each CONV layer exhibits kernels of $K^{(l)} \times q^{(l)}$. The CNN2D model can perform both spatial and spectral-spatial analysis by accepting spatial patches with few principal components or spectral-spatial patches with all (or most) available spectral bands, to which each CONV layer applies a kernel of $K^{(l)} \times k^{(l)} \times k^{(l)}$. Finally, the CNN3D model is employed for spectral-spatial analysis, taking full advantage of the spectral signatures contained in the input data by applying CONV layers with $K^{(l)} \times k^{(l)} \times k^{(l)} \times q^{(l)}$ kernels.

as input six PCs. Zhao et al. (2015) propose a a CNN2D architecture for extracting deep spatial features using, on the one hand, a multiscale convolutional AE based on the Laplacian pyramid and, on the other hand, the PCA to extract three PCs. Then, the extracted spatial features are concatenated together with the spectral information, using the logistic regression as a classifier. Furthermore, Ding et al. (2017) consider the HSI cube as a collection of 2-D images (i.e.images from different bands), which are cropped into patches to train a CNN2D model to automatically learn the data-adaptive kernels from the data through clustering.

In addition to introducing PCA-extracted spatial patches, some works propose the use of spatial-handcrafted features. For instance, Chen et al. (2017b) reduce the spectral domain to three PCs and extract spatial features (edges and textures) by applying Gabor Filtering. These features are sent to the CNN2D model, reducing the workload and addressing the overfitting problem. Another example is Romero et al. (2016), which performs a study between shallow and deep CNN2D models trained with the enforcing population and lifetime sparsity (EPLS) algorithm (Romero et al., 2015) for unsupervised learning of sparse multi/hyperspectral features.

Recently, a deformable HSI classification network model (DHCNet) has been proposed by Zhu et al. (2018a), using PCA to extract the three most informative PCs of the original HSI data cube and splitting the image into neighborhood windows to feed a CNN2D model, composed by deformable convolutions and downsampling that fuse the neighboring structural information of each input data sample in an adaptive manner.

4.4.3. Spectral-spatial CNN for HSI data analysis

Regarding spectral-spatial models, they consider both spectral properties and spatial information from the HSI data cube. In this sense, several strategies and architectures can be developed to perform the spectral-spatial processing, mainly due to the great flexibility that CNN models exhibit.

Following traditional pixel-wise methods, the CNN1D can be employed to perform spectral-spatial classification, rearranging the the spatial information and concatenating it to the spectral features (Zhang et al., 2016). For instance, Slavkovikj et al. (2015) integrate spatial and spectral information by reshaping the spectral-spatial neighborhood window to be processed by 1-D kernels, and Ran et al. (2017) improve the contextual information of the CNN1D by developing spatial PPFs (SPPFs), introducing the constraint that only the central pixel and its immediate surrounding pixels are paired.

Focusing on CNN2D architectures, these models can perform spectral-spatial processing in different ways. The most direct one is to feed the model with 3-D neighboring regions of size $d \times d \times n_{channels}$, where $n_{channels}$ can be certain number of PCs or the original n_{bands}. In this regard, some methods perform an initial DR in order to reduce the spectral correlation and redundancy. For instance, Makantasis et al. (2015) compose 3-D inputs with 10–13 PCs, applying the randomized PCA (R-PCA) over the HSI data cube. Yu et al. (2017) develop a spectral-spatial CNN with 1×1 CONV layers, also called cascaded cross-channel parametric pooling or CCCP layers (Lin et al., 2013a), and one global average pooling (GAP) layer instead of the traditional FC layers, to better analyze the HSI data information. Paoletti et al. (2017a)

develop a spectral-spatial model that efficiently takes into account the full spectrum, reaching competitive results, while Dong et al. (2019) propose a spectral-spatial CNN2D with a band-attention mechanism to improve the feature representation of the data.

The spectral-spatial processing can be performed by CNN2D architectures introducing spectral-spatial handcrafted features. For instance He et al. (2018) train the CNN2D model with covariance matrices, which encode the spectral-spatial information of different-sized neighborhoods of 20 PCs, obtaining multiscale covariance maps. Aptoula et al. (2016) use attribute profiles (APs) (Mura et al., 2010) as input to the CNN2D model, taking advantage of the spatial information and spectral properties that APs can capture in an image at various scales. Yue et al. (2015) develop a CNN2D architecture to process spectral and spatial features by composing the spectral information as three different feature maps, and concatenating them to the spatial patches (reduced by PCA to three PCs).

Moreover, several approaches combine the CNN2D with other different models to perform spatial and spectral feature extraction in separated fashion; for instance, Zhao and Du (2016) propose a spectral-spatial feature based classification (SSFC) approach that employs a CNN2D to find spatial-related features, while the spectral feature extraction is performed by a balanced local discriminant embedding algorithm (BLDE). Yue et al. (2016) extract spectral features from a SAE, while a multiscale spatial FE is performed by a CNN2D with spatial pyramid pooling (SPP). Zhang et al. (2017) and Yang et al. (2016) combine the hierarchical spectral and spatial-related features extracted from a CNN1D and CNN2D, respectively, performing the final classification with a softmax regression classifier. Ma et al. (2018b) introduce a two-branch model, where the spatial branch is composed by a CONV-DECONV architecture with skip connections, and the spectral branch is implemented by a contextual DNN.

In addition to the CNN1D and CNN2D models, the CNN3D model is usually adopted for spectral-spatial classification, where the 3-D filters of size $K^{(l)} \times k^{(l)} \times k^{(l)} \times q^{(l)}$ are able to extract high-level spectral-spatial features in a natural way, extracting as output $K^{(l)}$ feature volumes. For instance, Chen et al. (2016) review the three kinds of convolutional models that use the full pixel vectors in the original HSI data cubes to create the input blocks for their CNN3D model, and Li et al. (2017c) perform an interesting comparison between the spectral-spatial CNN3D model, two spectral-based methods (SAE and DBN), and the spatial CNN2D for HSI data classification, demonstrating that the CNN3D-based method is able to outperform these state-of-the-art methods. Furthermore, the CNN3D model can be used as a simple AE in order to obtain spectral-spatial features. For instance, Mei et al. (2019a) and Sellami et al. (2019) perform the classification on the spectral-spatial features obtained by a CNN3D model.

As with CNN1D and CNN2D models, the available literature offers more complex and sophisticated procedures for HSI data processing involving CNN3D architectures. For instance, Luo et al. (2018) develop a hybrid CNN2D-3D architecture able to deal with overfitting problems, using a 3-D kernel as the first layer of the network to extract feature vectors from the original 3D inputs, which are characterized by a small neighborhood window (only 3×3 with the full dimensionality given by n_{bands}). Then, the procedure reshapes the obtained feature vectors into one single matrix that is sent to the second 2-D kernel, and also to the subsequent pooling and FC layers, performing an end-to-end classification. With certain similarities, Leng et al. (2016) propose a cube-CNN-SVM (CCS) architecture which extracts several feature vectors from the original HSI data cube, performing the classification in an easy and efficient way with an SVM classifier. Roy et al. (2019) also combine a CNN3D with a CNN2D, where the CNN3D first extracts spectral-spatial features that are then refined by the CNN2D. Li et al. (2018b) follow a similar architecture, changing the final SVM by an RF. Moreover, Gao et al. (2018b) develop a CNN architecture with as many "branches" as AP features extracted from the HSI data cube, extracting independently the corresponding output volumes, which are concatenated and computed by the rest of the network. Cao et al. (2018) improve the performance of bayesian-inspired CNN3D model by placing spatial smoothness prior on data labels extracted with Markov random fields (MRFs) (Sun et al., 2015). Finally, Wang et al. (2019) introduce the alternately updated spectral-spatial CNN (AUSSC) as an end-to-end CNN3D with a recurrent feedback structure to learn refined spectral and spatial features.

4.4.4. Residual learning

CNN models have revolutionized the image processing field, establishing themselves as the current state-of-the-art. In this sense, the constant improvements in convolutional architectures, and their adoption in HSI data processing problems, have made possible to achieve performances never seen before in HSI classification (Khan et al., 2018). However, like the rest of DNN models, very deep CNN models must face some limitations related to the depth and the data degradation, as pointed out in Section 2.2. To overcome these issues, some works have focused on increasing the network's depth by creating short paths from low-level layers to high-level layers (i.e. residual connections). The development of convolutional architectures with residual learning has been a crucial step in the implementation of VDNN models, allowing the development of models with hundreds of layers.

The internal structure of residual neural networks (ResNets) (He et al., 2016) is based on groups of FE and detection stages which compose the basic building block, known as residual unit (Xie et al., 2017). The inputs of such block are directly connected to the outputs through an aggregation operation, as it can be observed in Fig. 8. Such residual connection performs an identity mapping that helps to propagate previous information to the subsequent units, improving the backward step by promoting the propagation of the gradient. In this regard, the output volume $\mathbf{X}^{(l)}$ of the l-th residual unit is given by Eq. (9), where $\mathcal{G}(\cdot)$ represents all the operations applied over the input data $\mathbf{X}^{(l-1)}$, which depend on all the parameters (weights $\mathcal{W}^{(l)}$ and biases $\mathcal{B}^{(l)}$) of the layers that compose the l-th unit:

$$\mathbf{X}^{(l)} = \mathcal{G}(\mathbf{X}^{(l-1)}, \mathcal{W}^{(l)}, \mathcal{B}^{(l)}) + \mathbf{X}^{(l-1)} \tag{9}$$

Eq. (9) reveals that the previous knowledge, in terms of generated features, is exploited once again in the current unit. The available literature gathers some works concerning the use of the ResNet in HSI processing. For instance, Zhong et al. (2017b) develop an end-to-end spectral-spatial ResNet (SSResNet) for HSI classification, outperforming traditional CNN models even with small training sets (Zhong et al., 2017c). Also, Paoletti et al. (2018c) present a pyramidal ResNet for spectral-spatial HSI data classification, improving the results of Zhong et al. (2017b). Lee et al. (2016) and Lee and Kwon (2017) propose the contextual deep CNN, which employs residual learning to simplify the training of the proposed network. Moreover, Mou et al. (2018) implement an unsupervised classification method based on the AE architecture with CONV-DECONV layers, following the ResNet architecture for spectral-spatial HSI data classification. In addition, Xie et al. (2018) and Yuan et al. (2019) employ the residual-based model as a spectral-spatial denosing AE for HSI data restoration and classification. Song

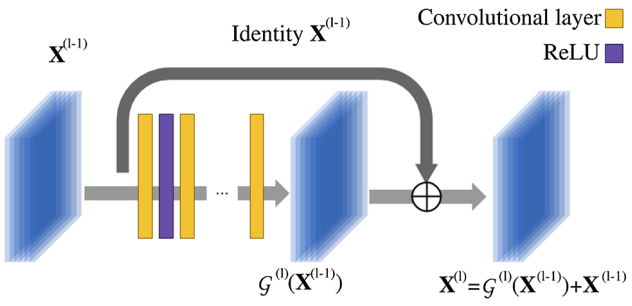


Fig. 8. Graphical visualization of a residual unit. The architecture reinforces the learning process of the model by reusing previous information.

et al. (2018) implement the deep feature fusion network (DFFN), composed by stages or branches of CONV layers connected with internal residual units, whose features are concatenated at the end (before the final classification). Li et al. (2019b) develop the multiscale deep middle-level feature fusion network (MMFN), an architecture that combines CONV layers and residual blocks into two stages to extract optimal multiscale features and to fuse and learn the complementary information from the obtained features. Chen et al. (2019a) present two DL ensemble methods based on CNNs and ResNets, implementing transfer learning to make full use of the learned weights. Moreover, recent works have focused on improving the performance of ResNets through attention techniques, for instance Haut et al. (2019) improve the spectral-spatial classification of the ResNet model including a visual attention mechanism to enhance the analysis of features. Mei et al. (2019b) develop a two-branch model, where the CNN's branch contains the spatial attention mechanism and the ResNet's branch implements the spectral attention mechanism. In addition to classification tasks, the ResNet has been also tested for HSI data super-resolution by Wang et al. (2017a), exhibiting good results.

The introduction of connections between different layers has inspired other models. Particularly, densely connected networks (DenseNets) (Huang et al., 2017) follow and extend the ResNet idea, reusing low-level, middle-level and high-level features by concatenating ($\frown$) all the previous feature maps obtained in a dense block (see Fig. 9). In this sense, the output of each dense block is calculated as the concatenation of the inner blocks that compose it:

$$\mathbf{X}^{(l)} = \mathcal{G}(\mathbf{X}^{(l-1)}, \mathcal{W}^{(l)}, \mathcal{B}^{(l)}) \frown \cdots \frown \mathcal{G}(\mathbf{X}^{(1)}, \mathcal{W}^{(1)}, \mathcal{B}^{(1)}) \tag{10}$$

In both cases, ResNets and DenseNets increase the number of connections, which does not imply a growth of model parameters that must be fine-tuned. Quite opposite, internal connections allow to reduce their number due to the presence of redundant information. At the same time, they reinforce the feature propagation along the network, performing a kind of regularization. Several works have adapted the DenseNet model to HSI processing tasks. For instance, Paoletti et al. (2018a) implement a Deep&Dense CNN model for spectral-spatial classification of HSI data, while Wang et al. (2018a) analyze spectral, spatial and spectral-spatial DenseNets for HSI classification. In a similar way to ResNet, the DenseNet can be combined with attention mechanims. For instance, Ma et al. (2019) propose the double-branch multi-attention mechanism network (DBMA) to separately extract spectral and spatial features, adopting (at each branch) an attention mechanism to extract the most discriminative features and Fang et al. (2019) propose an end-to-end 3-D DenseNet with spectral-wise attention mechanism for enhancing HSI classification. Also, the ResNet and DenseNet can be combined to construct a joint network, known as dual-path network (DPN) (Chen et al., 2017a), composed by bottleneck-blocks whose output is split into two branches: the first branch is element-wisely added to the residual path, and the second branch is concatenated with the densely connected path. This model has been successfully employed for HSI classification purposes; for instance, Kang et al. (2018) reach very good accuracy in comparison with ResNet and DenseNet, taking into account very small training sets (0.3%-0.5%) and using PCA to extract 5–10 PCs.

4.5. Other improved convolutional-based networks

In addition to ResNets and DenseNets, some other convolutional-inspired architectures have been developed for HSI data analysis. For instance, Liu et al. (2018) implement a siamese CNN (S-CNN) (Koch et al., 2015) for HSI data classification, which contains two branches of identical sub-networks that share the same configuration and parameters. This implies less parameters to fine-tune, requiring less training data and reducing the tendency to overfitting, helping to manage datasets with high intraclass variability and interclass similarity, reaching good performance with a small number of training samples.

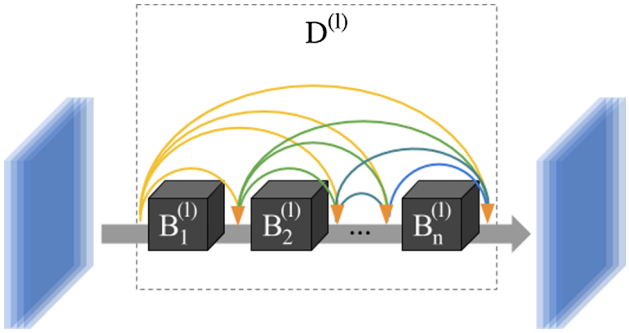


Fig. 9. Graphical visualization of a dense block. Instead of CONV layers, the DCNN is composed by dense blocks $D^{(l)}$, where each one contains several inner blocks $B_i^{(l)}$ composed by several CONV layers. The architecture of each $D^{(l)}$ allows for the reutilization of the low, middle and top feature maps extracted by each inner block.

Inspired by the inception architecture (Szegedy et al., 2015; Szegedy et al., 2016), the work of Lee et al. (2016) introduces an inception module at the beginning of their model, composed by n-parallel streams with several layers and different kernel sizes, whose outputs are merged by concatenation. Moreover, inspired by the network-in-network (NiN) (Lin et al., 2013a) architecture, Shamsolmoali et al. (2018) train a RNN with combined spectral-spatial features extracted by a CNNiN.

Based on the fully convolutional network (FCN) (Long et al., 2015a), whose learnable layers rely only on CONV and DECONV layers, Li et al. (2018a) implement an AE-based FCN for HSI-FE, using an ELM to classify the obtained features. Following the CONV-DECONV architecture and adding skip connections, the hourglass CNN architecture (Newell et al., 2016; Haut et al., 2018b) creates an encoder-decoder structure where each block of CONV layers that compose the encoder is connected to the corresponding DECONV layer at the decoder counterpart. This architecture can be employed for HSI data denoising (Sidorov and Hardeberg, 2019).

Recently, a new kind of network based on capsules and dynamic routing has been implemented, called Capsule Networks (CapNets) (Sabour et al., 2017). This architecture encodes the data internal relationships into an activity vector (instead of a traditional scalar value). Such data representation has demonstrated to be powerful in encoding useful features from the data, solving the limitations exhibited by the pooling layer. In this sense, Paoletti et al. (2018b) present a spectral-spatial CapsNet for HSI classification that outperforms the accuracies reached by traditional CNN models and the ResNet. Also, Deng et al. (2018) present a HSI-CapsNet that provides good results when very few training samples are employed.

5. Overcoming the limitations of DL in HSI Classification

The vast number of works discussing DNN models (in general) and CNNs (in particular) for HSI data analysis reveals the great possibilities that DL-based methods are able to offer in this context, not only in terms of architectural modifications, but also regarding their combination with other methods and algorithms, as we pointed out on Section 3. This also includes a large variety of remote sensing image processing techniques apart from data classification (FE, DR, unmixing, reconstruction, super-resolution, etc.) Convolutional-based networks such as CNNs and ResNets represent the most groundbreaking advance in DL in the last few years, allowing the implementation of VDNN models with hundreds/thousands of layers and compelling performance, following the assumption that deeper models are able to extract more complex and high-level relationships from the data (Srivastava et al., 2015). In the end, this expected to lead to improvements in model accuracy and performance (Krizhevsky, 2012; Yu et al., 2013). This has

also placed CNNs and ResNets as the current mainstream technologies in DL for HSI classification. However, these models must also face the limitations listed on Section 2, related to intrinsic problems of HSI analysis and the efficient management of depth. In order to deal with the aforementioned issues, several techniques and mechanisms have been developed in previous years to enhance the learning process and improve the performance of deep architectures. In the following, we provide a description of the available strategies to mitigate these issues.

5.1. Opening the black box

Regarding the "black box" nature of DNN models (in general) and convolutional-based ones (in particular), several efforts have been made to "open" the box and understand what are the filters actually doing (Rauber et al., 2017). For instance, *mNeuron* (Wei et al., 2017a) is a powerful Matlab plugin that allows the visualization of convolutional neural network parameters, while t-distributed stochastic neighbor embedding (t-SNE) (Maaten and Hinton, 2008) and uniform manifold approximation and projection (UMAP) (McInnes et al., 2018) are non-linear dimensionality reduction techniques which are also employed to visualize the models parameters in a simple way. Liu et al. (2017b) propose to formulate the CNN model as a directed acyclic graph (DAG), developing the CNNVis as a visual analysis system to better understand, diagnose and refine CNN models.

Regarding parameter visualization, several works propose to understand how DL is working in step-by-step fashion. In this context, Lei et al. (2018) present an ambitious dissertation, analyzing the DL-based models as physical systems from a microscopic, macroscopic, and worldview perspectives. Ravanelli and Bengio (2018) present a more concrete proposal, developing the SincNet, a convolutional-based model that exploits parametrized *sinc* functions in the first layer to discover more significant filters. Also, the BagNet (Brendel and Bethge, 2019) employs a visual bag-of-local-features model to perform the classification, extracting features that are easy to identify and interpret.

There is a wide variety of proposals to understand what networks do (Mahendran and Vedaldi, 2015; Nguyen et al., 2015). However, in the current literature about HSI-classification, little attention has been paid to this issue. Qiu and Jensen (2004) propose a method for understanding the performance of a three-layer MLP in HSI classification, but no relevant efforts have been reported with DL-architectures.

5.2. Reducing overfitting

In order to address the overfitting problem in convolutional-based models, several strategies have been reported that can be classified into four main categories: (i) those that affect the data, (ii) those that affect the model, (iii) those that affect the training process, and (iv) new learning paradigms to deal with the limited availability training data.

5.2.1. Data augmenting and noise inclusion

Gathering enough labeled samples to capture the high variability of HSI data is complicated, time consuming and expensive. Several works have focused on addressing this issue through the generation of virtual samples to enhance the robustness of convolutional-based models (Acquarelli et al., 2018). Following traditional methods, Yu et al. (2017) enlarge the training set by rotating and flipping the input spectral-spatial patches, while Lee and Kwon (2017) mirror the spectral-spatial training patches four times, across the horizontal, vertical and diagonal axes. On the other hand, Haut et al. (2019a) implement a mechanism to add spatial-structured noise to the input HSI data by randomly occluding some areas of the input patch in order to enhance the performance and robustness of the CNN model. Chen et al. (2016) present two methods to create additional training samples: the first one changes the spectral radiation of the original training samples $\mathbf{x}_i$ by multiplying them by a random factor and adding random noise, and the second one by mixing the spectral properties of two samples of the same class with proper ratios. Also, Acquarelli et al. (2018) present two methods: smoothing-based data augmentation, which takes advantage of the spectra of neighboring pixels, and label-based data augmentation, which exploits the labels of neighboring pixels to favor those classes with less samples, in addition to creating copies of the original data by inserting random noise. Ghamisi et al. (2016) propose a *dither* algorithm (Simpson, 2015) to suppress non-linear distortions and data aliasing, generating new samples by adding random noise to the original training samples, in addition to using the fractional order Darwinian particle swarm optimization (FODPSO) to select the most informative spectral bands. An interesting work has been recently proposed by He and Chen (2019), who implement a transformation network (STN) to obtain an optimal input of the CNN model for HSI classification, which translates, rotates and scales the network's input until obtaining an optimized one.

5.2.2. Reducing the complexity of the model

The second strategy to reduce overfitting is focused on reducing the computational complexity of deep CNNs (Maji and Mullins, 2018), for instance, by optimizing the internal structures of the CONV layers, pruning them to obtain a more simple and efficient network architecture (Cheng et al., 2017b). The thinning of the network (and the subsequent parameter reduction) make the CNN model lighter, which leads to faster training and execution, although not many efforts have been made in this direction in the HSI arena. Recently, works focused on designing optimal CNN architectures for HSI data processing have been presented. Specifically, Chen et al. (2019b) propose a methodology to automatically design efficient CNN1D and CNN3D architectures for HSI data classification. Given a number of operations (i.e., layers such as CONV, POOL or normalization), a gradient descent-based search algorithm evaluates all possible configurations and selects the best and optimal one.

5.2.3. Enhancing the training process

The methods for this purpose cover a wide range of techniques. For instance, L1/L2 regularization methods insert a penalty into the loss function in order to minimize the absolute value of the weights or the squared magnitude of the weights (weight decay or Tikhonov regularization), respectively (Murugan and Durairaj, 2017). The regularization process forces the model to make compromises on its weights, making it more general. In particular, the L1 regularization enforces the identification of the most relevant features in a dataset, while the L2 pursues a regularization that is less aggressive, but more efficient in computational terms. For instance, Chen et al. (2016) use the L2 regularization.

In addition to these methods, dropout regularization (Hinton et al., 2012; Srivastava et al., 2014) has also been proven to be a good solution to enhance the performance and robustness of the CNN model, preventing complex co-adaptations on training data. The mechanism is quite simple: it randomly deactivates a percentage of the activations in order to improve the network generalization, forcing the neurons to make more compromised assumptions. For instance, Paoletti et al. (2017a) make use of dropout in the layers of the CNN. Based on dropout, multiple regularization techniques have been developed (Ba et al., 2013; Zhang et al., 2016a; Molchanov et al., 2017) such as its generalization to large FC layers: the drop-connect (Wan et al., 2013), which sets randomly selected connection weights to zero. However, traditional dropout injects random single-pixel noise to the feature maps, resulting in spatially unstructured noise, which makes it ineffective in 2-D and 3-D models (Park and Kwak, 2017). In this sense, spatial-dropout (Tompson et al., 2015) and dropblock (Ghiasi et al., 2018) regularization techniques overcome the problem by dropping spatial-regions of the feature maps.

Another interesting regularization technique for preventing overfitting is early stopping (Caruana et al., 2001), which saves at each epoch those models that outperform the previous trained networks, discarding the others and storing at the end the results of the best

model. For instance Ran et al. (2017), Acquarelli et al. (2018), and Wang et al. (2018a) employ this technique to assess the convergence of their convolutional-based models.

5.2.4. Improvements on learning strategies

Overfitting in DNN models is intimately related to the number of samples available for training, representing one of the main limitations of supervised and very deep models. In this context, some improvements on learning paradigms have been developed in the DL field that can effectively improve the performance of DNNs when very few samples are available. We describe four of such paradigms: (i) semi-supervised and (ii) active learning (AL), (iii) transfer learning (TL), and (iv) self-supervised learning.

Semi-supervised learning In-between unsupervised and supervised learning, DNN models allow the implementation of hybrid approaches. In particular, semi-supervised learning (Ratle et al., 2010) provides a wide range of techniques to expand the training set $\mathcal{D}_{train}$ by including unlabeled data during the training stage (Ratle et al., 2010; Sabalel and Jadhav, 2014). For instance Ma et al. (2016c) present a semi-supervised learning strategy based on multi-decision labeling (local, global and self-decision levels), where unlabeled samples with high confidence are selected to extent the training set. Kang et al. (2019) extract pseudo-training samples from PCA and extended morphological attribute profiles (EMAPs) (Dalla Mura et al., 2011), applying extended random walker optimizers to feed a spectral-spatial convolutional-based deep feature fusion network (DFFN). Wu and Prasad (2018) present a convolutional-based recurrent model fed by pseudo training samples obtained by previous clustering. Fang et al. (2018) adopt separated spectral and spatial residual architectures with co-training, where the most confident labeled samples at each iteration are included in $\mathcal{D}_{train}$. A similar approach has been implemented by Zhou et al. (2019b), in which two separated spatial and spectral SAEs are co-trained, enlarging $\mathcal{D}_{train}$ by a region growing method. An interesting trend is the adoption of the ladder network (Rasmus et al., 2015; Pezeshki et al., 2016), a new DNN model based on hierarchichal latent variable models, for semi-supervised classification of HSI data (Liu et al., 2017a; Büchel and Ersoy, 2018).

In addition to the addition of unlabeled data to $\mathcal{D}_{train}$, some semi-supervised techniques are able to replicate new samples. In particular, the DNN structure known as generative adversarial network (GAN) (Goodfellow et al., 2014). For instance, Zhu et al. (2018b) propose convolutional-based GAN1D and GAN3D architectures to learn the intrinsic characteristics of HSI data, enhancing the classification performance achieved by the traditional CNN1D and CNN3D. Also He et al. (2017b), Zhu et al. (2018b), and Zhan et al. (2018) present similar approaches, while Zhang et al. (2018a) introduce a Wasserstein GAN to perform unsupervised FE.

Active learning (AL) AL is a semi-supervised machine learning algorithm (MacKay, 1992) that can easily deal with the availability of a limited amount of labeled data by training the model with a small set of labeled samples that is reinforced by the acquisition of new (representative and intelligently selected) unlabeled samples, reducing the cost of acquiring large labeled training sets and the number of needed training samples. In the literature, several works combining the DNN and AL paradigms can be found. For instance, Haut et al. (2018c) discuss the use of Bayesian CNNs for spectral, spatial and spectral-spatial HSI classification, providing robust classification results in comparison with traditional CNN models. In addition to convolutional models, Liu et al. (2017c) employ AL with a DBN, while Li (2015) develops an AL-based SAE.

Transfer learning (TL) The TL paradigm (Yosinski et al., 2014; Long et al., 2015b) is based on the assumption that learned features in one task can be used for other tasks (Pan and Yang, 2010). Low-level layers in convolutional-based architectures are able to learn generic features that are less dependent on the final task, while the top-level layers learn more specific knowledge, extracting features that are more related with the final task. In this sense, TL-based algorithms usually employ off-the-shelf pre-trained networks (i.e. models that were trained on different datasets) to process the data of interest, tailoring them slightly for the new task by removing some the last few layers of the model and retraining again with some new final layers. As a result, the amount of data used to pre-train the CNN can be leveraged, alleviating the need for new data (this is useful when limited amounts of training sets are available) and producing better results in a shorter amount of time (Windrim et al., 2018). The most widely used off-the-shelf pre-trained networks are trained with the ImageNet dataset, composed by 14 million images belonging to 1000 different classes. The most popular topologies are the following ones:

- *ResNet-50*, composed by 50 residual layers,
- *DenseNet121* (Huang et al., 2017), composed by 4 dense blocks connected by transition layers,
- *VGG-16* and *VGG-19* (Simonyan and Zisserman, 2014), which increase the depth by using many layers: 16 and 19, respectively, using a simple architecture with small kernels (CONV layers of 3×3) and reducing the volume size (through POOL layers of 2×2),
- *MobileNet* (Howard et al., 2017), whose architecture is suitable for onboard processing, maximizing the accuracy while taking into account restricted resources for an integrated application,
- *Xception* (Chollet, 2017), based on inception networks, where the original modules have been replaced with depthwise separable convolutions in order to make a more efficient use of model parameters.

Several works adapt the TL paradigm to process HSI data (Mei et al., 2017; Jiao et al., 2017; Windrim et al., 2018; Yang et al., 2017; Deng et al., 2019; Zhang et al., 2019), visibly improving the training of deep CNN models when limited amounts of labeled data are available for the training stage.

Self-supervised learning This learning strategy emerges as an alternative approach to supervised learning, being able to extract the naturally available contextual information and embedded metadata as supervisory signals, without an explicit need for $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{n_{labelled}}$ pairs. This does not mean learning the inherent structure of data in the form of unsupervised learning (Liu et al., 2019; Jing and Tian, 2019). In HSI data classification, Wang et al. (2018b) propose the HSINet, which contains a three-layer DNN, a multi-feature CNN, and an embedded conditional random field to achieve self-supervised feature learning, extracting spatial, spectral, color, boundary and contextual information. Also, Liang et al. (2018) combine TL with self-supervised learning, developing a pre-trained VGG-16 to extract deep multi-scale spatial information from the HSI data cube, whose spatial information is processed together with the raw spectral information by a SAE.

5.3. Vanishing gradient problem

Apart from improvements based on architectures, such as ResNets and CapsNets, those methods employed to deal with the vanishing gradient problem (Srivastava et al., 2015) can be categorized into three main groups: (i) implementing data normalization between each network layer, (ii) developing better initialization strategies with proper optimizers, and (iii) implementing better non-linear activation functions.

5.3.1. Avoiding vanishing gradient through data normalization

During gradient descent training, the layer's weights $\mathbf{W}^{(l)}$ and the obtained data $\mathbf{X}^{(l)}$ distributions can vary (covariate shift effect), making the learning very unstable and saturating the activations whose first derivative tends to zero. This leads to the vanishing gradient problem. In this sense, it is common to employ normalization methods to control the magnitude and mean of the neurons' activations located into one layer (independently of the other layers of the model). This aims at

performing the parameters' optimization in an easier way (Santurkar et al., 2018) while, at the same time, dealing with the unbounded nature of certain activation functions (for instance, the ReLU), whose outputs are not constrained within a bounded range (such as the tanh function). Table 2 provides a summary of several relevant normalization methods that have been adopted in this context.

5.3.2. Avoiding vanishing gradient through initialization and optimization strategies

Classical DNNs initialize their parameters, setting small random values to the weights and biases that compose the model, under the assumption that this helps the stochastic optimization algorithm used to train the model. In this sense, the selection of the optimization algorithm becomes fundamental in order to obtain a proper performance of the model. This selection must take into account the type of data to be used, the task to be performed, and the features of the problem.

Several optimization methods with different strengths and weaknesses have been developed with the aim of improving the process of minimizing an objective function: the traditional stochastic gradient descent (SGD), which is faster than standard gradient descent (GD) but harder to converge to the minimum due to frequent updates and fluctuations; the minibatch SGD, which reduces the high variance in the parameter updates; the momentum, which speeds SGD by descending along the relevant direction, reducing oscillations; the preconditioned SGD (PSGD) (Li, 2018), which adaptively estimates a preconditioner for handling efficiently the gradient noise and non-convexity of a target function at the same time, giving good results in deep neural models optimization (Li et al., 2016); Adagrad (Duchi et al., 2011), a variant of PSGD which adapts the learning rate based on the parameters, being well-suited for dealing with sparse data although its learning rate can suffer from constant decaying, producing the *decaying learning rate* problem and hampering the optimization process; AdaDelta (Zeiler, 2012), which tries to avoid the decaying learning rate problem by calculating different learning rates for each parameter and is often combined with the momentum technique; and finally the adaptive moment estimation (Adam) (Kingma and Ba, 2014), which is a combination of Adagrad and AdaDelta, outperforming the previous optimization techniques. It is based on processing adaptive learning rates for each parameter and storing several past gradients to keep a decaying average of those past gradients, which makes it efficient, with fast convergence and effective when dealing the vanishing learning rate. The excellent results of Adam algorithm have positioned it as the method that is most widely used for optimizing deep networks, being employed in some HSI-related works as Paoletti et al. (2017a, 2018c).

In addition to the improvements implemented on the optimizers (Martens et al., 2012; Sutskever et al., 2013; Dauphin et al., 2014), recently new investigations have been made in order to improve the initialization of model parameters (Bengio et al., 2007a; Glorot and Bengio, 2010; Erhan et al., 2010; He et al., 2015; Koturwar and Merchant, 2017; Guo and Zhu, 2018), for instance by performing unsupervised pre-training (Romero et al., 2016; Li et al., 2015a), which initializes the parameters near to a local minimum, allowing for a better generalization via unsupervised FE.

5.3.3. Avoiding vanishing gradient through new non-linear activation techniques

Currently, several efforts for preventing the vanishing-gradient problem have been made based on developing effective non-linear activation functions $\mathcal{H}(\cdot)$ (Xu et al., 2015a; Pedamonti, 2018). In particular, some rectified-based activation functions have been adapted to overcome the problem by preventing the gradient from being zero. For instance, in order to face the dying ReLU problem the *leaky* ReLU (LReLU) (Maas et al., 2013) and *parametric* ReLU (PReLU) (He et al., 2015) functions have been implemented with Eq. (11) (see Fig. 3).

$$\mathcal{H}(x) = \begin{cases} a \cdot x & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases} \tag{11}$$

In particular, the LReLU sets the gradient signal as a linear component of the input layer data $\mathbf{X}^{(l)}$, employing a small and constant negative slope (usually $a = 0.001$) when the data is equal or smaller than 0. This avoids the *dying* ReLU problem, as the function will not have zero-slope parts, making the LReLU more balanced and allowing a faster learning. PReLU works similarly, being a learnable in this case. In this context, the vanishing gradient depends on the slope a. Instead of that, the *scaled exponential linear unit* (SeLU) (Klambauer et al., 2017; Paoletti et al., 2018) derives two parameters: α and λ from the inputs, as we can observe in Eq. (12), allowing $-\lambda\alpha$ as the smallest gradient value and mapping the means and variances from one layer to the next one in order to minimize the covariate shift effect.

$$\mathcal{H}(x) = \lambda \begin{cases} \alpha e^x - \alpha & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases} \tag{12}$$

6. Popular deep learning frameworks

The current trend in the literature is to implement deeper and more complex networks, with new topologies, more branches and connections, and better optimizers and functions. In this sense, certain programming frameworks have been deployed in order to provide technical and coding support to developers of DL methods. In particular, these DL frameworks offer a black-box environment for training and validating DNN models through a high level programming interface. Furthermore, instead of *ad hoc* software, the framework provides quality and maintainability of applications at low cost, allowing for the model to better adapt to market standards. In terms of performance, available frameworks are able to easily exploit computing tools, developed and supported by large communities, relying on well-known high performance computing (HPC) libraries such as CUDA, CUDNN, MKL, BLAS, AVX operations and Protobuf, among others.

Table 3 provides a summary of the main DL frameworks currently available in public repositories, including a brief description, the programming language that was used for coding purposes, and the available application interfaces (APIs). It is interesting to highlight the use of Python as one of the main programming languages in the community to implement DL frameworks, due to its versatility and flexibility. In addition, the number of stars and forks have been provided as indirect evaluation metrics of those repositories (see Fig. 10), where stars measure the degree of popularity of the repository, while forks measure the number of copies that have been made of the original repository. These data has been obtained on two different dates: July 16th, 2018 and September 8th, 2019, in order to compare the evolution of these indicators. It can be observed that the most popular DL framework is TensorFlow (Abadi et al., 2016a; Abadi et al., 2016b), tracked by more than 120.000 followers and with more than 70.000 branches and forks, being the framework that has grown the most from 2018 to 2019. Concerning TensorFlow, the high-level library Keras (Ketkar, 2017) has also experimented an increase in the number of followers, allowing for the development of ANN models in an easy and simple way. Also, it is interesting to note that the frameworks based on Torch (Collobert et al., 2002), Pytorch (Fey and Lenssen, 2019) and Fast.AI have also significantly grown, providing an easy-to-debug tool for the implementation of neural models.

7. Experimental results

7.1. Hyperspectral datasets

After reviewing the main models and frameworks, we perform a comparison between the most popular DL-based architectures and traditional ML-based algorithms in order to quantify the improvements

Table 2
Some examples of normalization methods for neural networks.

Method	Description
Local response normalization (LRN) (Krizhevsky et al., 2012)	It was introduced by the first time in the AlexNet model to enhance the *lateral inhibition* property of the neurons, i.e. the ability of the neural nodes to reduce the activity of its neighbors by competition, modulating the feedback signals and enhancing the visual contrast (i.e. performing an attention mechanism) (Wyatte et al., 2012). In this sense, the LRN allows to diminish responses that are uniformly large for the neighborhood, making large activation more pronounced within a neighborhood and creating higher contrast in activation maps in order to increase the sensory perception. It has been successfully applied by (Lee et al., 2016; Lee and Kwon, 2017).
Batch normalization (BN) (Ioffe and Szegedy, 2015)	Considering the output volume $\mathbf{X}^{(l)}$ of the l-th layer as the data to normalize, with n_{batch} data representations, where each one comprises $K^{(l)}$ feature maps of size $d^{(l)} \times d^{(l)} \times n^{(l)}_{channels}$, being n_{batch} the batch size, $d^{(l)} \times d^{(l)}$ the spatial dimensions and $n^{(l)}_{channels}$ the number of spectral bands, the BN method normalizes the obtained features by computing the mean μ and variance σ^2 respect to the feature maps (channel) dimension. It has been widely used by the HSI community (Liu et al., 2017a; Zhong et al., 2017c; Gao et al., 2018a; Deng et al., 2018), being usually applied before the activation function, to maintain the data distribution to zero-mean and unit variance, scaling and shifting the data through the learnable parameters γ and β, respectively. This allows to reach a more independent and high-speed learning (with larger learning rates and high accuracy), although it is very sensitive to the batch size n_{batch} (Bjorck et al., 2018).
Weight normalization (WN) (Salimans and Kingma, 2016)	It normalizes the weights of the l-th layer, reparameterizing $W^{(l)}$ in terms of a parameter vector $\mathbf{v}$ (weights' direction $\mathbf{v}/\|\mathbf{v}\|$) and a scalar parameter g (weights' norm, which is also obtained by a learnable log-scale parameter s as $g = e^s$) and directly performing the backpropagation with respect to those parameters instead, in order to fix the Euclidean norm of $\mathbf{W}^{(l)}$. This, coupled with the mean-only batch normalization, allows the scale of neural activations to be approximately independent of the parameter $\mathbf{v}$, as well as their mean (Gitman and Ginsburg, 2017).
Layer normalization (LN) (Ba et al., 2016)	Similar to BN, LN normalizes the data computing the mean $\mu_{\mathcal{B}}$ and variance $\sigma^2_{\mathcal{B}}$ with respect to the batch dimension, in order to avoid the limitations of the BN method. In this sense, LN does not employ batch statistics, being the normalization of each sample independent of other samples. This enables a beneficial behaviour in networks such as RNNs.
Instance normalization (IN) (Ulyanov et al., 2016b)	Inspired by Huang and Belongie (2017) in neural style transfer tasks Ulyanov et al. (2016a), the IN normalizes across each feature map dimension in the batch independently avoiding the dependency of the batch and normalizing the constrast of the content image. It is commonly used to remove variance of images on low-level vision tasks (Pan et al., 2018). Also, coupled with BN, IN has inspired the development of other methods, such as batch-instance normalization (BIN) (Nam et al., 2018) that extends the handling of the variability introduced by visual styles (textures, lighting, filters) to general recognition problems.
Group normalization (GN) (Wu and He, 2018)	GN divides the feature map (channel) dimension into several groups, normalizing each group in the current batch, exhibiting a behavior that straddles the layer and instance normalization methods depending on the number of groups that it creates.
Batch re-normalization (BRN) (Ioffe, 2017)	It extends the BN method in order to deal with small or non-independent and identically distributed (non-i.i.d) batches, normalizing the activations through the combination of the batch's mean and variance ($\mu_{\mathcal{B}}$ and $\sigma^2_{\mathcal{B}}$, respectively) and the moving averages (μ and σ^2) in an affine transformation.
Decorrelated batch normalization (DeBN) (Huangi et al., 2018)	BN is able to scale and shift the obtained activations through parameters γ and β. In this sense, the DeBN extends the BN method to perform data whitening, taking into account the zero-phase component analysis (ZCA) method (Kessy et al., 2018) to decorrelate the neural activations.

Table 3
Some of the most widely used DL-based frameworks (data obtained on September 8th, 2019).

NAME	DESCRIPTION	APIs	STARS	FORKS
Tensorflow	An Open Source Machine Learning Framework for Everyone	C++, Go, Java, JavaScript, Python, Swift	133322	77067
Keras	Deep Learning for Humans	Python	43786	16673
OpenCV	Open Source Computer Vision Library	C++, Java, Python	37798	27983
PyTorch	Tensors and Dynamic Neural Networks in Python with Strong GPU Acceleration	C++, Python	31416	7712
Caffe	A fast open framework for deep learning	CLI, Matlab, Python	29000	17530
MXNet	Lightweight, Portable, Flexible Distributed/Mobile Deep Learning with Dynamic, Mutation-aware Dataflow Deep Scheduler	C++, Clojure, Java, Julia, Perl, Python, R, Scala	17647	6276
CNTK	Microsoft cognitive toolkit (CNTK), an Open Source Deep-learning Toolkit	C++, C#, Python	16394	4365
Fast.AI	The Fast.ai Deep Learning Library, plus Lessons and Tutorials	Python	15384	5517
Deeplearning4j	Eclipse Deeplearning4j, ND4J, DataVec and more - deep learning & linear algebra for Java/Scala with GPUs + Spark	Java/Scala	11130	4735
Paddle	PArallel Distributed Deep LEarning	Python	9851	2628
ConvNetJS	Deep Learning in Javascript. Train CNNs (or ordinary ones) in your browser.	Javascript	9787	1951
Theano	Python library that allows to define, optimize, and evaluate efficiently mathematical expressions involving multi-dimensional arrays	Python	8900	2504
Horovod	Distributed training framework for TensorFlow, Keras, PyTorch, and Apache MXNet	Python	7380	1130
Chainer	A flexible Framework of Neural Networks for Deep Learning	Python	5028	1327
BigDL	BigDL: Distributed Deep Learning Library for Apache Spark	Python/Scala	3152	797
MatConvNet	CNNs for MATLAB	MATLAB	1205	713

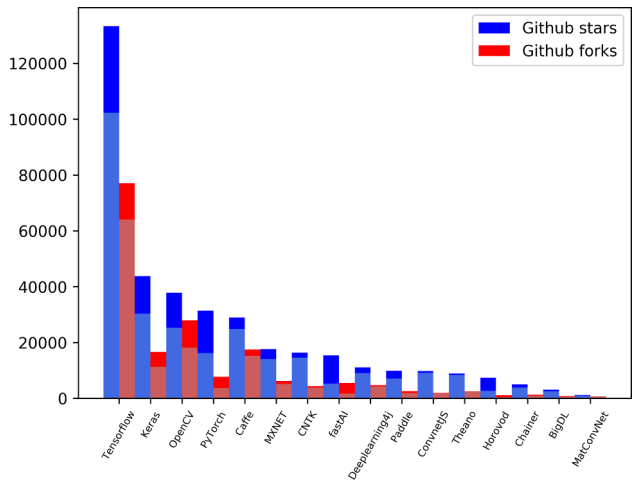


Fig. 10. Stars and forks of the most representative DL framework repositories. The light blue and red bars refer to the number of stars and forks measured on July 16th, 2018, while the dark blue and red bars correspond to the number of stars and forks measured on September 8th, 2019 (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.).

and advantages that can be gained by DL models in terms of performance and classification accuracy. To this end, four images widely used in the field of hyperspectral image processing have been selected to complete the experimental part of the work: the Indian Pines (IP) and Salinas Valley (SV) scenes, collected by AVIRIS, the University of Pavia (UP) scene, gathered by ROSIS and the University of Houston (UH) scene, collected by CASI. Table 4 shows a brief summary of these HSI datasets, including the number of labeled samples per class, as well as the available ground-truth information:

- The IP dataset (Table 4) was captured in 1992 by the AVIRIS sensor (Green et al., 1998) over the Indian Pines test site in NW Indiana, an agricultural area characterized by its crops of regular geometry and also irregular forest regions. The scene consists of 145×145 pixels with a spatial resolution of 20 mpp and with 224 spectral bands, which have been collected in the wavelength range from 0.4 to 2.5 μm. From these bands, 24 were removed for being null or water absorption bands (in particular [104–108], [150–163] and 220), considering the remaining 200 bands for the experiments. The ground truth available is divided into sixteen classes and about half of the data (10249 pixels from a total of 21025) contains labeled samples.
- The UP scene (Table 4) was acquired by the ROSIS sensor (Kunkel et al., 1988) over the campus of the University of Pavia, in the north of Italy. The dataset contains nine different classes that belong to an urban environment with multiple solid structures, natural objects and shadows. After discarding the noisy bands, the considered scene contains 103 spectral bands, with a size of 610×340 pixels with spatial resolution of 1.3 mpp and covering the spectral range from 0.43 to 0.86 μm. Finally, about 20% of the pixels (42776 of 207400) contain ground-truth information.
- The SV image (Table 4) was gathered by the 224-band AVIRIS sensor over several agricultural fields of Salinas Valley, California, and it is characterized by a spatial resolution of 3.7 mpp. The area covered comprises 512×217 spectral samples. As in the case of the IP dataset, we discard 20 bands due to water absorption and noise.
- The UH scene (Xu et al., 2016a) was collected by CASI in June 2012 over the University of Houston campus and the neighboring urban area. This scene forms a cube of dimension $349 \times 1905 \times 144$, with spatial resolution of 2.5 m and spectral information captured in the range from 0.38 to 1.05 μm, containing 15 ground-truth classes divided in two categories: training (top UH map in Table 4) and and testing (bottom UH map in Table 4). In this sense, the UH scene provides an interesting benchmark dataset, which was first presented at the IEEE Geoscience and Remote Sensing Society (GRSS) Image Analysis and Data Fusion Technical Committee during the 2013 Data Fusion Contest (DFC) (Debes et al., 2014).

These datasets, along with the training and test data, are all available online from the GRSS Data and Algorithm Standard Evaluation (DASE) website (http://dase.grss-ieee.org).

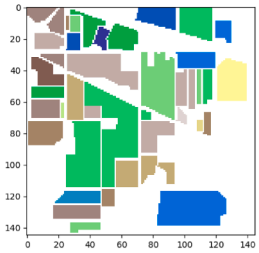
7.2. Experimental settings

In order to make an exhaustive analysis of the main DL-based architectures employed for HSI classification purposes, an extensive set of experiments have been carried out.

1. The first experiment compares the performance of supervised standard ML and DL classification methods with different amounts of training samples over the four considered HSI datasets, studying how they are affected by the lack of information and the type of samples. In particular 1%, 5%, 10%, 15%, 20% and 25% of the labeled samples per class have been randomly selected to compose the training set on IP, UP and SV, while the full available training set for UH has been considered. Also, some of the most popular classification algorithms available in the literature have been considered: (1) random forest (RF), (2) multinomial logistic regression (MLR), (3) support vector machine (SVM) with radial basis function kernel (Waske et al., 2010), (4) multilayer perceptron (MLP), (5) vanilla recurrent neural network (RNN), (6) RNN with gated recurrent unit (GRU), (7) RNN with long short term memory (LSTM), (8) spectral CNN (CNN1D), (9) spatial CNN with 2-D kernels and one PC (CNN2D), (10) spectral-spatial CNN with 2-D kernels and forty PCs (CNN2D40), and (11) spectral-spatial CNN with 3-D kernels and also forty PCs (CNN3D). Regarding the configuration of the experiment, the available training set has been divided into batches of 100 samples, using Adam optimizer with learning rate of 0.0008 for SV and UP and 0.001 for IP and UH. Regarding the number of epochs, MLP, CNN1D, CNN2D and CNN2D40 have been trained using 300 epochs. The parameters of RNN, GRU and LSTM models have been adjusted using 200 epochs. Finally, the parameters of CNN3D have been trained using 100 epochs. Furthermore, the topology details of each model are reported on Table 5. It must be noted that we follow the convention that deep architectures have at least two or more hidden layers, while shallow models are composed by single-hidden layer architectures (Bengio et al., 2007b; Schmidhuber, 2015). In addition, the inclusion of batch normalization (BN) in some layers of the convolutional models intends to, on the one hand, avoid vanishing/exploding gradients and, on the other hand, maintain the distribution of the layer's inputs (internal covariate shift) (Ioffe and Szegedy, 2015). We have empirically observed that, on some (but not all) CNN models, BN stabilizes and accelerates the training stage. In this sense, we noted that these configurations helped these particular convolutional models. Furthermore, we also empirically observed that a filter size of 5×5 provided better results than traditional kernels of 3×3 (widely used in VGG-16 and similar architectures).
2. The second experiment performs a specific comparison between CNN models with and without handcrafted features. In this sense, the IP, UP and UH datasets have been considered, employing as spectral-spatial model the CNN baseline, the CNN with extended morphological profiles (EMP-CNN) and the CNN with Gabor filtering (Gabor-CNN) proposed by Ghamisi et al. (2018), whose architectures are composed by two feature extraction and detection stages, where each one contains a stack of CONV-ReLU-POOL layers. These have been fed with input patches of 27×27 pixels, preserving

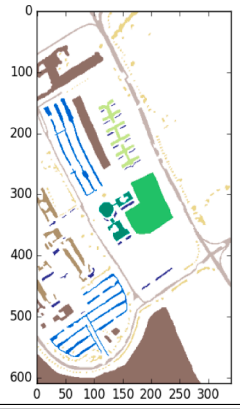
Table 4
Number of available samples in the Indian Pines (IP), University of Pavia (UP), Salinas Valley (SV), and the University of Houston (UH) datasets. The samples for the latter scene are divided in two categories: training (top) and testing (bottom).

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)

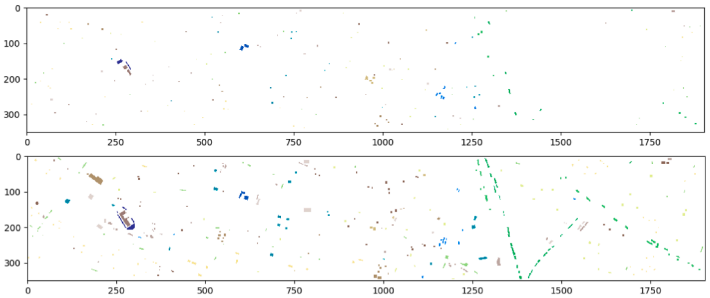


Color	Land-cover type	Samples
	Background	164624
	Asphalt	6631
	Meadows	18649
	Gravel	2099
	Trees	3064
	Painted metal sheets	1345
	Bare Soil	5029
	Bitumen	1330
	Self-Blocking Bricks	3682
	Shadows	947
	Total samples	207400

SALINAS (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

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

three PCs for EMP-CNN and Gabor-CNN models and the full spectrum for the CNN baseline. Also, 50 samples per class have been considered (when using the IP scene) to train the models, and 548, 540, 392, 524, 256, 532, 375, 514, and 231 labels of each class (see Table 4) have been employed for testing the UP scene, while for the UH scene all available training samples have been considered.

3. The third experiment compares the performance of several improved convolutional-based architectures, in particular residual and capsule-based models, over two HSI datasets, using 20% and 10% of the available labeled samples for IP and UP datasets, respectively. We have considered five deep architectures: (1) the spectral-spatial residual network (SSRN) (Zhong et al., 2017b), (2) the spectral-spatial pyramidal residual network (P-RN) (Paoletti et al., 2018c), (3) the densely connected CNN (DenseNet) (Paoletti et al., 2018a), (4) the spectral-spatial dual-path network (DPN) (Kang et al., 2018), and (5) the capsule network (CapsNet) (Paoletti et al., 2018b). Moreover, with the aim of exploring the performance of these methods with different levels of spatial information, four different spatial neighborhoods have been tested: 5×5, 7×7, 9×9 and 11×11.

Table 5
Neural network base model topologies considered in our experiments, emphasizing the input, hidden and output layers in order to demonstrate the depth of each architecture. In this sense, the term "*linear input*" refers to the input layer of each model, while the last densely-connected layer with sofmax function is the output layer. Regarding the input layer, spectral models receive pixel-vectors of n_{bands} elements, while spatial and spectral-spatial methods employ an input patch size of $19 \times 19 \times n_{channels}$, being $n_{channels} = 1$ for CNN2D and $n_{channels} = 40$ for CNN2D40 and CNN3D. Finally, the term "*recurrentLayer*"(†) indicates that this layer has been implemented by a RNN/GRU/LSTM layer, depending on the kind of neural network. The number in the parentheses indicates the number of units (i.e. the dimensionality of the layer).

Model	Main layer	Norm.	Ac. Function	Downsampling
MLP	Linear input(n_{bands})	–	–	–
	FC($n_{bands} \cdot \frac{2}{3} + 10$)	–	ReLU	–
	FC(n_{class})	–	Softmax	–
RNN	Linear input(n_{bands})	–	–	–
GRU	recurrentLayer†(64)	–	Tanh	–
LSTM	recurrentLayer†(64)	–	Tanh	–
	FC(n_{class})	–	Softmax	–
CNN1D	Linear input(n_{bands})	–	–	–
	CONV(20×24)	–	ReLU	POOL(5)
	FC(100)	BN	ReLU	–
	FC(n_{class})	–	Softmax	–
CNN2D	Linear input($19 \times 19 \times n_{channels}$)	–	–	–
	CONV($50 \times 5 \times 5$)	–	ReLU	–
	CONV($100 \times 5 \times 5$)	–	ReLU	POOL(2×2)
CNN2D40	FC(100)	BN	ReLU	–
	FC(n_{class})	–	Softmax	–
CNN3D	Linear input($19 \times 19 \times n_{channels}$)	–	–	–
	CONV($32 \times 5 \times 5 \times 24$)	BN	ReLU	–
	CONV($64 \times 5 \times 5 \times 16$)	BN	ReLU	POOL($2 \times 2 \times 1$)
	FC(300)	BN	ReLU	–
	FC(n_{class})	–	Softmax	–

4. The fourth experiment studies how semi-supervised techniques (in particular, the AL paradigm) are affected by the amount of training data available when combined with DL-models, in particular spectral, spatial and spectral-spatial convolutional-based models, taking into account four classifiers with Bayesian perspective (Haut et al., 2018c): (1) AL-MLR, (2) spectral CNN (CNN1D), (3) spatial CNN with 2-D kernels and input patches keeping one PC with PCA (CNN2D) and (4) spectral-spatial CNN with 3-D kernels and input patches keeping all the spectral bands of the original datasets (CNN3D). The IP and SV datasets have been considered for this experiment.
5. The fith experiment performs two comparisons to analyze the performance of different TLapproaches. Specifically, the first one performs a comparison between five off-the-shelf deep models, studying their classification accuracies over three HSI datasets: IP, UP and SV, and employing the TL paradigm. In this sense (1) VGG16 (Simonyan and Zisserman, 2014), (2) VGG19 (Simonyan and Zisserman, 2014), (3) ResNet50 (He et al., 2016), (4) MobileNet (Howard et al., 2017), and (5) DenseNet121 (Huang et al., 2017) have been considered. These models have been pre-trained with the ImageNet, followed by a general training using IP, UP and SV datasets in order to fit their BN layers, using Adam optimizer, a learning rate of 0.0001 and 5 epochs. Then, a hidden FC layer with 256 neurons and an output FC layer with $n_{classes}$ neurons have been added at the end of these models, which were fine-tuned employing several training percentages (1%, 5%, 10%, 15%, 20% and 25%). This fine-tuning is carried out with Adam optimizer, a learning rate of 0.001 and 50 epochs. In addition, a second comparison is carried out, comparing the performance of the CNN1D, CNN2D, CNN2D40 and CNN3D models implemented in our first experiment (see Table 5) employing the TL paradigm. In this sense, IP and SV have been considered because of their spectral similarities, as the two scenes were collected by the same spectrometer (AVIRIS). First, these models have been pre-tained using the IP scene, because of its spectral complexity, and then tested over the SV scene. The model parameters are adjusted with 2, 4, 8, 16, 32, 64, 128 and 256 samples per SV class.
6. All previous experiments have been developed by randomly selecting the training data from the available set of labeled samples (with the exception of UH scene, which employs its own set of fixed training samples). In this context, new trends suggest that the high correlation between neighboring pixels can affect the performance of the network, in the sense that the test set will be very close to the train set, allowing the model to obtain too optimistic results, which are not adjusted to the real generalization power of the model. Regarding this, our sixth experiment compares the performance of the models considered on the first experiment (i.e. RF, MLR, SVM, MLP, RNN, GRU, LSTM, CNN1D, CNN2D, CNN2D40 and CNN3D) trained with spatially disjoint samples of IP and UP datasets (these training and test sets are available from the GRSS DASE website at http://dase.grss-ieee.org).

In order to assess the results of these experiments, three widely used quantitative metrics are used to evaluate the classification performance: (i) the *overall accuracy* (OA), that computes the number of correctly classified HSI pixels divided by the number of samples, (ii) the *average accuracy* (AA), that computes the mean of the classification accuracies of all classes, and (iii) the *Kappa coefficient*, that measures the agreement between the obtained classification map and the original ground-truth map.

All our experiments have been conducted on a hardware environment composed by a 6th-generation Intel R Core TM i7-6700 K processor, with 8 MB of Cache and a processing speed of 4.20 GHz with 4 cores/8 way multi-task processing. It includes 40 GB of DDR4 RAM with 2400 MHz serial speed and a Toshiba DT01ACA hard disk with 7200RPM and 2 TB capacity. The environment is completed with a NVIDIA GeForce GTX 1080 graphics processing unit (GPU) with 8 GB GDDR5X video memory and 10 Gbps memory rate, and an ASUS Z170

pro-gaming motherboard. The software environment consists of the Ubuntu 18.04.1 x64 operating system with CUDA 9.0 and cuDNN 7.1.1 and Python 2.7 as the programming language.

7.3. Experimental discussion

7.3.1. Comparison between standard supervised HSI classifiers and DL-based networks

Our first experiment intends to compare different supervised classifiers, analyzing how the training percentage affects their performance. In this sense, the considered methods can be separated into two broad categories: traditional ML-based methods (RF, MLR, SVM, MLP) and DL-based networks (RNN, GRU, LSTM, CNN1D, CNN2D, CNN2D40 and CNN3D). Also, a second categorization can be made by dividing the proposed methods into spectral classifiers (RF, MLR, SVM, MLP, RNN, GRU, LSTM, CNN1D), spatial classifiers (CNN2D), and spectral-spatial classifiers (CNN2D40 and CNN3D).

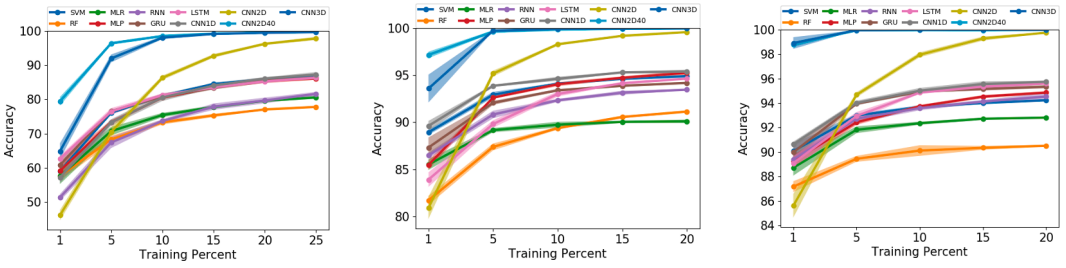
Fig. 11 gives the obtained results for IP, UP and SV datasets after the execution of five Monte-Carlo runs. It is interesting to analyze the behavior of traditional ML methods when few labeled samples are available: they are highly affected by the lack of training data, being the MLP the one exhibiting the best performance and RF the worst, in general. Also, the pixel-based DL classifiers: vanilla RNN, GRU, LSTM and CNN1D are highly affected by the limited availability of training samples, exhibiting a similar behavior with regards to SVM and MLP, with slightly higher accuracy when enough training data are employed. In this case, we highlight the more stable performance of the CNN1D. Regarding the spatial classifier (CNN2D), it presents the worst accuracy when few samples are used in the training stage, even below traditional ML methods, although the spatial information when using patches of size 19×19 seems to be sufficient to reach a remarkably good accuracy with 15%-25% of training. Although the use of spatial information is highly effective (with a suitable training percent), the conjunction of spatial and spectral information achieves the best classification results. In this sense, the CNN2D40 and the CNN3D are able to achieve an OA near 100% with only 5% of training data in the considered datasets, being the IP the hardest scene to classify in our opinion. If we compare CNN2D and CNN2D40, we can observe how the spectral information is able to reduce the uncertainty of the classifier when few training data is available. In addition, the 2-D kernels of CNN2D40 classifier allows to reduce the overfitting in comparison with the 3-D kernels of the CNN3D, reaching similar results when enough training data are available.

Figs. 12–14 and Tables 6–8 present detailed classification maps and accuracy measures for IP (15% of training), UP (10%) and SV (10%) scenes. As we can observe, the spectral classifiers exhibit the familiar *salt and pepper* noise (significantly less in the DL-based methods), because they ignore spatial-contextual information when providing a pixel prediction. On the contrary, spatial and spectral-spatial classifiers exhibit more regular results, with less noise at the edges. However, the spatial CNN2D results often degrade some object and material shapes, a problem that is considerably reduced with the spectral-spatial CNN3D, providing classification results that are more similar with regards to the corresponding ground-truth maps for IP, UP and SV datasets. In addition, Tables 6–8 indicate the runtime of each considered method, being the standard ML-based classifiers the fastest ones (in particular, the SVM) although the consumed time during their parameter search has not been reflected, and the CNN-based algorithms the slowest ones due to the computational complexity of the CONV layers. Moreover, we can observe the number of parameters that each neural model needs to adjust during the training phase, being the MLP the model with fewest parameters and the CNN3D the one with the most parameters to fit.

Also, in order to provide a detailed comparison with a HSI benchmark, Table 9 and Fig. 15 show the classification results of considered methods over the UH dataset, employing the available training data to adjust the parameters of each supervised model. As we can observe in Table 9, spectral classifiers (RF, MLR, SVM, and MLP, RNN, GRU, LST and CNN1D) are able to reach good accuracies: between 73–87% of OA, with the CNN1D being the best pixel-wise classifier, because its kernel is able to process the spectral signatures in a more robust way than tradidional ML models and FC architectures of neural-inspired models. However, if we focus on the spatial classifier (CNN2D), we can see that it exhibits the worst OA, AA and Kappa values. This behaviour may be due to the fact that the reduction of the spectrum to a single band can generate samples that are very mixed and difficult to discriminate. In this sense, the available training samples are less descriptive for setting the parameter values, and they become insufficient for the 378015 parameters of the spatial model. In this sense, the spectral information is the key to discriminate correctly the samples of the UH dataset, as it can be observed in the spectral-spatial CNN2D40. Although this model has 48750 parameters more than its spatial counter-part, the CNN2D40 is able to take into account the original spectral information in its spatial features, obtaining feature maps that are more representative of the input data and being 1.86 times better than those provided by the spatial CNN2D. Furthermore, the 3-D kernels of the spectral-spatial CNN3D model are able to process these spectral features, combining them with the spatial information in order to obtain the output volumes. The classification maps in Fig. 15 demonstrate that spectral classifiers are very noisy, being in general unable to classify the area hidden by the cloud in the UH scene, while the CNN3D reaches a better result in general (see the parking areas, for instance) and showing some spatial structures of the hidden area under the cloud, such as buildings and parking lots.

7.3.2. Comparison between convolutional models, with and without handcrafted features

Our second experiment compares the performance of: (i) a classic spectral-spatial CNN for HSI classification (Ghamisi et al., 2018), which receives as input data patches of size $27 \times 27 \times n_{bands}$ extracted from the original cube, (ii) a spatial CNN that processes extended morphological profiles (EMP-CNN) obtained from the HSI data (Ghamisi et al., 2018) (using input patches of $27 \times 27 \times 3$), and (iii) a spatial CNN that



(a) Classification on IP (b) Classification on UP (c) Classification on SV

Fig. 11. OA evolution (y-axis) of each considered classifier with different training percentages (x-axis) over IP, UP, SV datasets. The standard deviation is also shown around each plot.

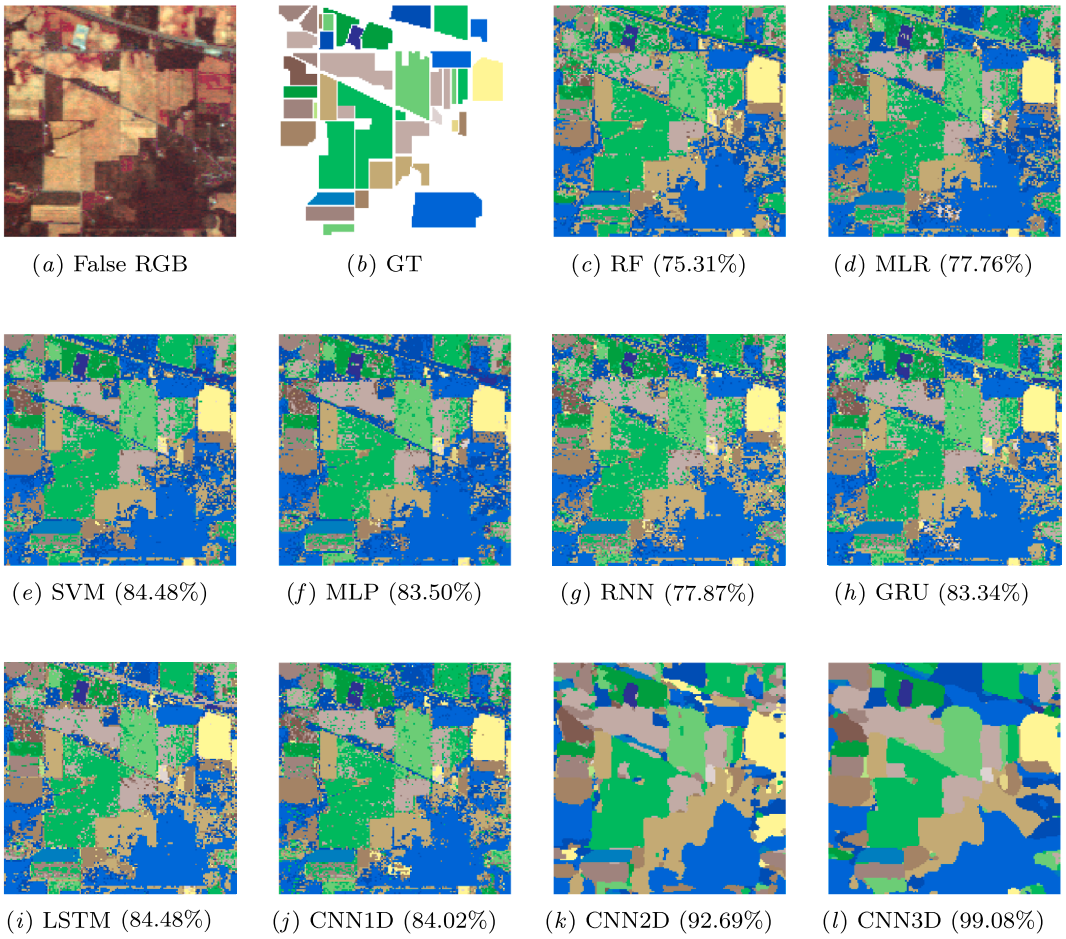


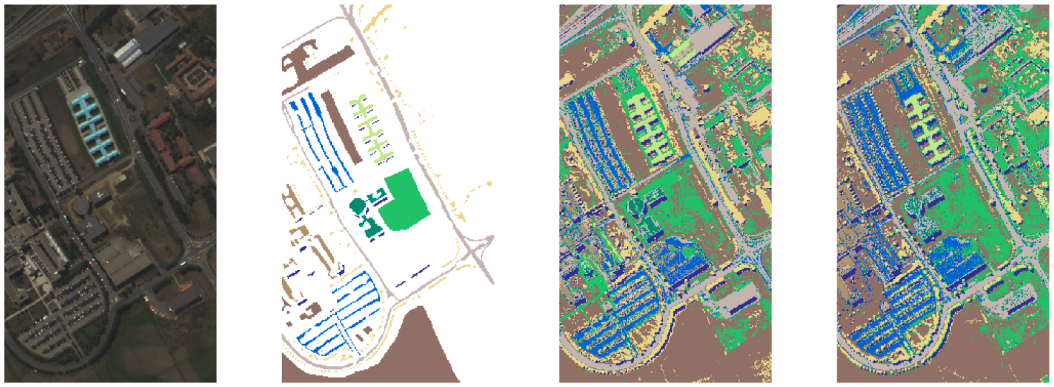
Fig. 12. Classification maps for the IP dataset with 15% of training data. Images from (a) to (j) provide the classification maps corresponding to Table 6. The corresponding overall classification accuracies (OAs) are shown in brackets.

processes the Gabor filtered data (Gabor-CNN) (Ghamisi et al., 2018), also employing input patches of size $27 \times 27 \times 3$, in order to observe the effects of extracting deep features directly from the data or from handcrafted features.

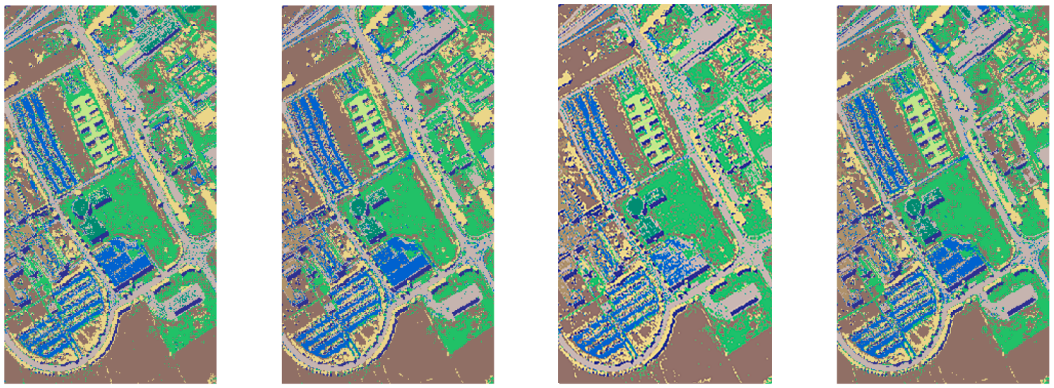
The results obtained over three HSI datasets: IP, UP and UH, are reported on Table 10. If we focus on the CNN baseline, the obtained results are in line with those shown in Tables 6, 7 and 9. Comparing the baseline with EMP-CNN and Gabor-CNN models, it is easy to confirm that the Gabor-CNN model exhibits the best performance for all the considered datasets, with the EMP-CNN being slightly worse. In this context, the CNN-baseline appears to provide the worst results in this particular case, with two to four percentage points below the Gabor-CNN. With these results in mind, we highlight that spatial-based processing of the data by powerful pre-processing methods, such as EMPs and Gabor filters, can significantly improve the performance of convolutional models. Particularly, Gabor filters exhibit optimal localization properties in both the spatial and frequency domains, allowing for the successful combination of spatial and spectral information for the extraction of edges and textures, while EMPs are also quite effective in the task of modelling the spatial-contextual information contained in the HSI data cube. This confirms and extends the obtained results of previous works, such as the one by Anwer et al. (2018), where explicit texture descriptors (local binary patterns) are used to improve classification results on several pre-trained models and aerial remote sensing benchmarks with RGB images.

7.3.3. Comparison between improved convolutional-based architectures

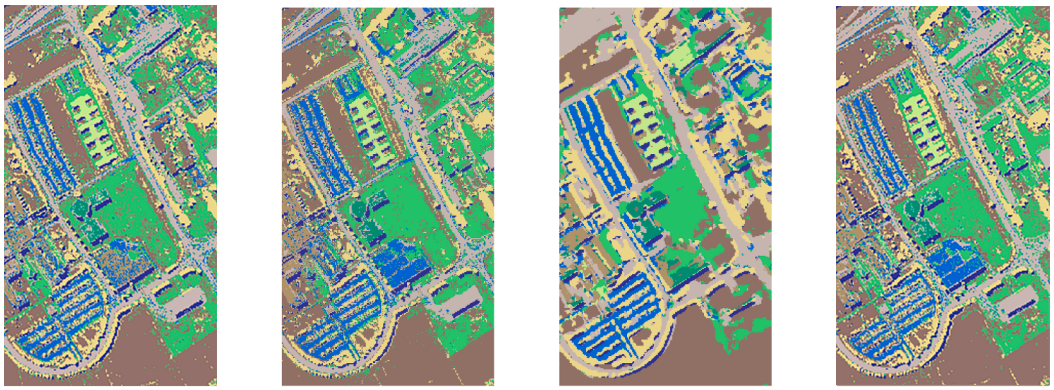
Our third experiment performs a study about the performance of improved convolutional-based models, considering different levels of spatial information. In this sense, it must be noted that these architectures have been particularly developed to efficiently exploit their depth, to obtain deeper and more abstract features, while avoiding the problems associated with the depth through communication mechanisms that reuse the data of the model, such as residual connections or dynamic routing. Table 11 shows the obtained results. As we can observe in the table, these methods are able to reach good accuracy, with small-sized patches being able to reach the 99% of OA using patches of size 7×7. In addition, although the SSRN and the P-RN use the same residual learning approach, the selection of the topology and the residual block architecture can substantially improve the performance of the network. In particular, the SSRN implements two networks (both with two residual units): one spectral network with all its kernels of size $1 \times 1 \times 7$, and one spatial network with all its kernels of size $3 \times 3 \times 128$. This reduces the number of parameters but prevents the efficient extraction of spectral-spatial information. However, the P-RN introduces only one network with three pyramidal residual modules, each one composed by three pyramidal bottleneck residual units, implementing its CONV layers of kernels 1×1, 7×7 and 8×8. Although the P-RN is more complex and deep than the SSRN, its performance is significantly better. At the end, the topology allows the P-RN to achieve significant precision gains, especially with smaller input spatial sizes. Also, it is interesting to highlight the standard deviation of both classifiers, which is lower in the P-RN model. The residual block architecture of P-RN is able to extract additional feature maps (as the residual units become deeper) in comparison with the SSRN, exploiting better the information contained within HSI input patches. In the end, this improves the OA results and reduces the standard deviation, i.e. the uncertainty.



(a) False RGB (b) GT (c) RF (89.37%) (d) MLR (89.73%)



(e) SVM (94.10%) (f) MLP (94.04%) (g) RNN (92.32%) (h) GRU (93.39%)



(i) LSTM (93.0%) (j) CNN1D (95.61%) (k) CNN2D (98.96%) (l) CNN3D (99.92%)

Fig. 13. Classification maps for the UP dataset with 10% of training data. Images from (a) to (j) provide the classification maps corresponding to Table 7. The corresponding overall classification accuracies (OAs) are shown in brackets.

Looking at the results obtained by DenseNet and CapsNet, these classifiers exhibit very similar behavior, reaching accuracy values between those obtained by the SSRN and P-RN when small spatial patches are used as input data (maintaining significant quantitative improvements with respect to the other HSI classifiers in the previous experiment), and even outperforming the results obtained with a high amount of spatial-contextual information. Finally, if we compare the residual models (SSRN and P-RN) and the DenseNet with the DPN model, we can observe that the DPN is able to outperform the results obtained by the SSRN with few training samples, while its accuracy is normally between that achieved by the P-RN and the DenseNet.

7.3.4. Comparison between semi-supervised and AL models

Our fourth experiment explores the use of labeled data during the training stage in AL models, with the aim of analyzing how the amount of training data affects their performance. In this context, the considered models follow a Bayesian perspective (Haut et al., 2018c), where each one extracts probabilistic information about the samples, in order to select those samples that provide more information to the model while, at the same time, reducing the number of training samples.

The obtained results over IP and SV datasets are shown in Table 12. As we can observe, the AL-CNN3D is able to reach 99% OA with approximately 3.92% of labeled data from IP and 0.53% of labeled data

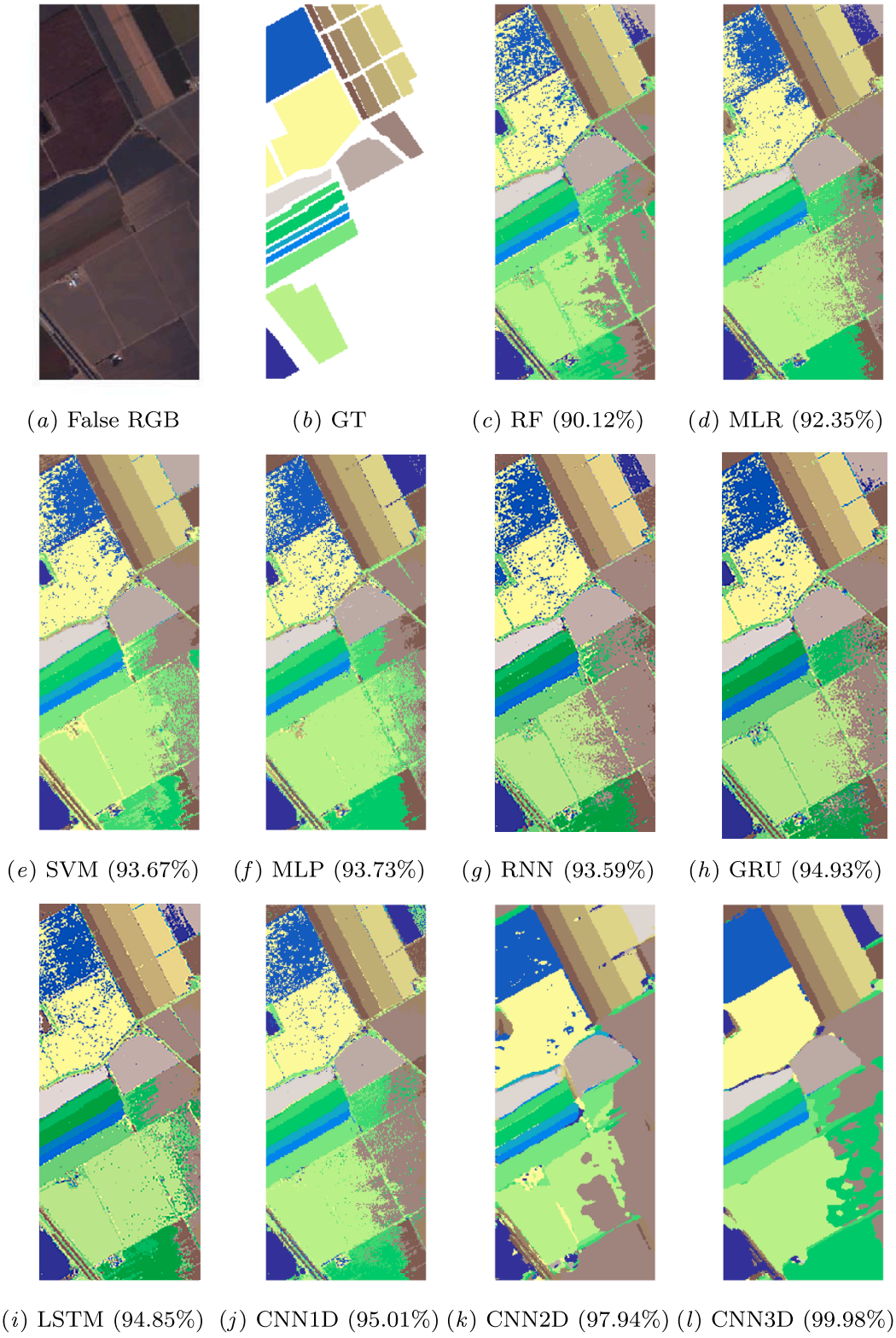


Fig. 14. Classification maps for the SV dataset with 10% of training data. Images from (a) to (j) provide the classification maps corresponding to Table 8. The corresponding overall classification accuracies (OAs) are shown in brackets.

from SV, which is in line with the results obtained in previous experiments: IP exhibits higher complexity compared with SV, whose pixels are spectrally less mixed and the spatial distribution is more geometric, with bigger areas made up of crops. On the contrary, the AL-MLR and AL-CNN1D are unable to reach such high OAs. For instance the AL-CNN1D is not able to improve 90% OA with the IP dataset, and it also cannot reach 99% OA with the SV scene. Furthermore, although the AL-CNN2D classifier is able to reach 99% OA in all the considered HSI scenes, it generally needs more labeled data than its spectral-spatial counterpart. These results strongly support the fact that joint spectral-spatial features are more useful than separate spatial and spectral features, making the AL-CNN3D model ideal for the extraction of highly

Table 6
Classification results for the IP dataset using 15% of the available labeled data.

Class	RF	MLR	SVM	MLP	RNN	GRU	LSTM	CNN1D	CNN2D	CNN2D40	CNN3D
Alfalfa	20.00	32.82	62.05	50.77	36.92	57.43	80.51	44.61	75.38	95.39	**96.92**
Corn-notill	61.53	75.07	81.45	78.90	73.49	80.17	82.19	81.04	91.54	98.73	**98.91**
Corn-min	53.62	57.96	70.55	66.27	58.04	70.33	70.16	70.69	86.95	**98.95**	98.84
Corn	35.12	45.67	72.93	61.19	44.28	66.47	53.83	60.10	88.56	**99.50**	97.71
Grass/Pasture	84.39	86.98	93.17	89.61	86.98	88.83	89.76	92.34	86.05	98.58	**99.32**
Grass/Trees	96.10	96.36	97.32	96.55	96.97	95.84	96.77	97.29	96.13	99.06	**99.74**
Grass/pasture-mowed	29.57	47.83	84.35	75.65	53.91	75.65	81.74	69.57	82.61	**93.91**	93.04
Hay-windrowed	96.11	99.16	98.32	97.54	98.67	98.67	98.52	98.18	97.88	100.00	100.00
Oats	1.18	18.82	51.76	61.18	27.06	68.23	62.35	44.70	65.88	98.82	**100.00**
Soybeans-notill	65.96	66.54	77.87	78.18	67.41	78.86	76.78	78.67	89.85	99.15	**99.15**
Soybeans-min	89.13	79.53	85.10	86.10	80.09	81.87	83.13	83.42	95.28	**99.62**	99.23
Soybean-clean	46.59	58.25	79.09	78.85	65.56	81.11	80.75	83.97	88.65	97.14	**97.86**
Wheat	92.18	98.51	98.39	98.74	97.93	98.51	98.62	98.62	97.82	99.77	**99.89**
Woods	94.53	95.31	95.59	94.55	92.11	95.35	93.58	94.51	98.40	**99.87**	99.59
Bldg-Grass-Tree-Drives	40.55	63.90	61.28	65.55	65.18	64.21	67.44	67.44	89.21	**99.45**	98.48
Stone-steel towers	83.54	85.06	87.60	89.37	86.08	86.58	82.78	87.59	82.53	**96.20**	95.70
OA	75.31	77.76	84.48	83.50	77.87	83.34	83.48	84.02	92.69	**99.14**	99.08
AA	61.88	69.24	81.05	79.31	70.67	80.51	81.18	78.30	88.29	98.38	**98.40**
K(x100)	71.41	74.46	82.26	81.13	74.65	80.98	81.13	81.75	91.65	**99.02**	98.95
Parameters				31047	217296	242640	255248	72616	378116	426866	1805196
Time (s.)	1.29	6.05	**0.25**	26.46	63.59	47.22	53.36	53.91	59.28	103.76	185.07

Table 7
Classification results for the UP dataset using 10% of the available labeled data.

Class	RF	MLR	SVM	MLP	RNN	GRU	LSTM	CNN1D	CNN2D	CNN2D40	CNN3D
Asphalt	91.63	92.39	94.29	93.81	92.33	94.33	93.02	95.85	98.01	99.97	**100.00**
Meadows	97.71	96.09	97.49	97.58	97.08	96.98	97.01	98.13	99.41	99.98	**100.00**
Gravel	66.88	73.27	80.84	78.11	75.43	77.63	78.18	81.48	93.90	**99.43**	99.35
Trees	89.10	86.90	94.21	93.59	91.89	94.04	94.14	94.15	98.14	99.32	**99.74**
Painted metal sheets	98.60	99.59	99.22	99.52	99.49	99.44	99.54	99.82	99.57	100.00	100.00
Bare Soil	64.35	77.83	90.91	91.64	87.2	88.14	86.4	91.71	98.08	99.99	**100.00**
Bitumen	77.66	56.34	87.35	85.53	82.07	84.88	86.77	87.52	89.72	99.80	**99.98**
Self-Blocking Bricks	88.52	86.68	87.47	88.92	84.38	88.37	87.27	85.68	98.28	99.61	**99.74**
Shadows	99.74	99.67	99.86	99.53	99.7	99.67	99.79	**99.88**	98.87	98.33	99.60
OA	89.37	89.73	94.10	94.04	92.32	93.39	93.0	94.61	98.27	99.83	**99.92**
AA	86.02	85.41	92.40	92.02	89.95	91.5	91.35	92.69	97.11	99.60	**99.82**
K(x100)	85.67	86.27	92.17	92.09	89.79	91.22	90.7	92.84	97.71	99.78	**99.89**
Parameters				8823	71817	97161	109769	33909	377409	426159	1803089
Time (s.)	4.29	8.63	**0.44**	68.22	150.06	113.07	128.09	139.58	139.82	226.22	448.32

discriminative features for classification purposes.

7.3.5. Comparison between transfer learning approaches

In the first test of our fifth experiment, we compare the performance of five off-the-shelf DL-based models, which have been pre-trained over the ImageNet and fine-tuned with different training percentages for the IP, UP and SV datasets.

The obtained results are given in Fig. 16. Focusing on the IP dataset, it can be observed that, with only 1% of labeled samples, the best OA is reached by the DenseNet121, which achieves 67.80% OA, being closely followed by the MobileNet. In this regard, it must be noted that IP dataset exhibits higher complexity than the UP and SV scenes, where the best results with 1% of the available labeled samples used for training are achieved by DenseNet121 with 94.37% and 98.02%, respectively. However we must highlight that, although these deep and very deep models have not been specifically developed for HSI analysis, they are able to reach interesting results in comparison with those obtained by the specially-designed CNN models in the first experiment (see Section 7.3.1). For instance, if we focus on the IP scene, VGG16, Resnet50, MobileNet and DenseNet121 models are able to outperform the CNN2D model with 1% of training data, while MobileNet and DenseNet121 outperform the results obtained by CNN1D and CNN2D40. With the UP dataset, VGG16, Resnet50, MobileNet and DenseNet121 models outperform the OA of CNN1D and CNN2D models, while with the SV dataset all pre-trained models outperform the CNN2D's results, and VGG16, Resnet50, MobileNet and DenseNet121 improve the classification accuracy of the CNN1D.

When more labeled samples are used for training, the OA increases quite fast. For instance, with 5% of training data, the vast majority of classifiers are able to reach at least 90% OA in the IP scene, and 99% in the UP and SV scenes, with few exceptions (for instance the VGG19 with the IP scene). Compared with the previous results reported on Section 7.3.1 we can observe that, in general, pre-trained models are slightly worse than the CNN2D40 and the CNN3D. In addition to the architectural design of the models, it must be highlighted that the spatial size and the spectral resolution of the input patch is decisive in improving the behavior of these deep networks. In this case, all the TL-based models have been fed with patches of size $32 \times 32 \times 3$, which are then scaled to the original inputs of the networks (for instance, the VGG16 employs patches of $224 \times 224 \times 3$). This limitation forces us to reduce the spectral dimensionality with PCA, which leads to a reduced capacity for spectral discrimination (while employing an excessively

Table 8
Classification results for the SV dataset using 10% of the available labeled data.

Class	RF	MLR	SVM	MLP	RNN	GRU	LSTM	CNN1D	CNN2D	CNN2D40	CNN3D
Brocoli green weeds 1	99.46	99.47	99.63	99.57	99.48	99.67	99.42	99.88	99.45	99.90	**100.00**
Brocoli green weeds 2	99.83	99.94	99.91	99.87	99.91	99.94	99.9	99.96	99.51	99.96	**100.00**
Fallow	99.15	98.60	99.68	99.44	99.1	99.58	99.65	99.85	99.62	100.00	100.00
Fallow rough plow	99.42	99.28	99.31	99.25	98.36	99.52	99.39	99.57	**99.89**	99.84	99.86
Fallow smooth	97.87	99.12	99.35	99.09	98.56	99.33	99.37	99.05	99.88	99.88	**99.95**
Stubble	99.68	99.92	99.80	99.85	99.8	99.86	99.89	99.85	99.78	100.00	100.00
Celery	99.39	99.89	99.54	99.57	99.71	99.75	99.7	99.84	99.64	100.00	100.00
Grapes untrained	84.42	87.98	90.51	86.88	87.22	89.83	90.79	90.98	95.60	99.96	**99.97**
Soil vinyard develop	99.07	99.73	99.92	99.73	99.77	99.79	99.76	99.83	99.54	100.00	100.00
Corn senesced green weeds	91.56	95.79	97.71	96.56	96.49	97.56	96.63	98.03	98.45	99.94	**99.99**
Lettuce romaine 4wk	94.13	95.90	98.88	97.81	97.59	98.52	98.86	98.33	98.73	99.94	**100.00**
Lettuce romaine 5wk	98.79	99.63	99.79	99.65	99.46	99.87	99.63	99.96	99.58	99.99	99.99
Lettuce romaine 6wk	97.86	99.03	98.88	99.03	98.45	98.66	99.05	99.17	99.13	**100.00**	99.98
Lettuce romaine 7wk	91.34	96.03	97.65	96.80	96.82	98.09	97.61	97.34	97.53	99.88	**99.98**
Vinyard untrained	60.46	66.63	70.54	77.81	76.79	80.98	79.59	79.52	95.01	**99.96**	99.95
Vinyard vertical trellis	97.06	98.89	99.18	99.08	98.95	99.07	98.7	99.00	97.00	99.94	**99.94**
OA	90.12	92.35	93.67	93.73	93.59	94.93	94.85	95.01	97.94	99.96	**99.98**
AA	94.34	95.99	96.89	96.87	96.65	97.5	97.37	97.51	98.65	99.95	**99.98**
K(x100)	88.98	91.47	92.94	93.02	92.86	94.35	94.27	94.44	97.71	99.96	**99.98**
Parameters				32282	221392	246736	259344	74616	378116	426866	1805196
Time (s.)	2.85	65.21	**0.94**	86.63	191.62	152.44	163.35	177.78	177.29	282.69	551.72

large spatial size).

At this point, it is important to note that the use of TL-based approaches in image processing tasks has two main benefits: the ability to achieve good results with few training samples extracted from the target scene and the reduction of runtime in the training procedure. However, although TL-based approaches are fairly reliable when few labeled samples are available, the models employed for HSI classification are based on those trained by the DL community over RGB datasets, such as ImageNet. In this sense, the effectiveness of TL methods depends mostly on the source application with which the models were pre-trained, and on the relationship with the final target application in which they will be used (Patricia and Caputo, 2014). In such case, the Imagenet dataset is not related with the employed HSI data and, hence, it was expected that these models would not be able to exhibit their full potential in HSI classification.

To overcome this limitation, in our second test we study the performance of the proposed CNN1D, CNN2D, CNN2D40 and CNN3D models implemented on the first experiment (see Table 5) employing the TL paradigm over two HSI datasets: IP and SV. In this context, the pursued goal is to take advantage of TL's ability to learn general knowledge from other datasets and then apply such knowledge to a specific task, polishing it on the target scene. Regarding this goal, we take advantage of the most spectrally mixed and difficult samples from IP to later recognize more precise characteristics in SV, employing 100% of the labeled data from IP scene (i.e. 10249 samples) to perform the pre-training stage, while extracting 2, 4, 8, 16, 32, 64, 128 and 256 samples from SV scene to adjust the considered models. The obtained results are given in Fig. 17. As we can observe, the behaviour is very similar for each model. In other words, pre-training with IP labels allows the considered models to achieve better accuracy when they are inferring the SV samples with very few training samples. However, the improvement is less significant when additional labeled samples from SV are added to adjust the model parameters, which demonstrates that (broadly speaking) TL is only recommended when there are very few

Table 9
Classification results for UH dataset using the fixed training set available.

Class	RF	MLR	SVM	MLP	RNN	GRU	LSTM	CNN1D	CNN2D	CNN2D40	CNN3D
Grass healthy	82.49	**82.62**	82.34	81.58	82.19	82.24	82.05	81.75	61.12	80.48	81.79
Grass stressed	83.36	83.93	83.36	81.67	83.44	81.35	81.56	**95.04**	50.08	85.49	87.20
Grass synthetic	97.82	99.80	99.80	99.64	99.84	99.88	99.76	99.88	29.35	88.99	94.73
Tree	91.74	98.01	**98.96**	88.69	94.64	96.14	91.89	89.45	46.61	83.66	84.74
Soil	96.80	97.16	98.77	97.08	97.99	97.12	97.56	98.63	41.36	**100.00**	99.81
Water	99.16	94.41	97.90	94.41	95.24	**99.3**	96.5	95.94	44.06	92.59	97.76
Residential	75.28	74.25	77.43	76.79	**81.05**	77.76	78.1	80.88	61.14	74.65	76.56
Commercial	33.01	65.15	60.30	55.82	42.72	48.4	39.79	80.32	32.95	80.85	**81.06**
Road	69.40	69.12	76.77	69.91	79.28	74.96	77.94	77.09	59.43	81.34	**88.46**
Highway	43.86	54.44	61.29	49.71	48.86	61.64	48.17	72.57	32.45	63.69	**78.30**
Railway	70.36	76.09	80.55	75.67	74.84	80.91	77.53	86.36	44.42	93.74	**96.28**
Parking lot1	54.77	73.39	79.92	77.16	74.99	81.73	81.4	91.91	33.68	96.96	**98.91**
Parking lot2	60.14	68.42	70.88	72.21	69.61	69.4	71.02	74.74	**84.00**	82.88	72.56
Tennis court	98.87	98.79	100.00	99.03	100.0	99.92	99.43	99.36	68.67	98.79	97.90
Running track	97.50	95.98	96.41	**98.31**	97.29	97.76	97.25	98.14	15.69	97.34	96.36
OA	73.09	79.53	81.86	77.98	78.44	80.39	78.11	86.66	45.80	85.18	**87.95**
AA	76.97	82.10	84.31	81.18	81.46	83.23	81.33	88.14	47.00	86.76	**88.83**
K(x100)	71.09	77.89	80.43	76.29	76.75	78.79	76.46	85.53	41.53	83.90	**86.91**
Parameters				16975	150735	176079	188687	50515	378015	426765	1804895
Time (s.)	2.68	21.25	**0.37**	46.09	105.81	78.45	88.90	94.41	81.69	165.33	311.56

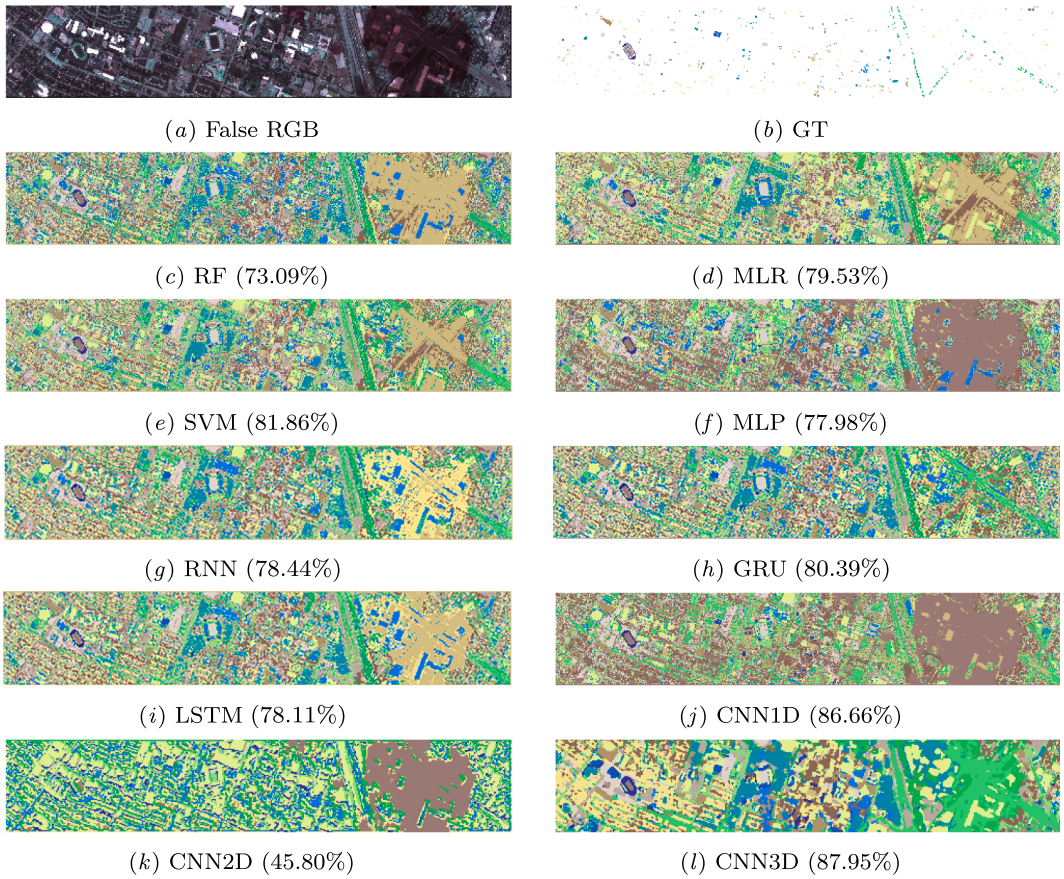


Fig. 15. Classification maps for the UH dataset. Images from (a) to (j) provide the classification maps corresponding to Table 9. The corresponding overall classification accuracies (OAs) are shown in brackets.

samples in the target scene and there is another (larger) dataset with similar characteristics that can help to model the parameters of a classification network in a reasonable way.

7.3.6. Training and testing with spatially disjoint samples

As it can be observed on Figs. 18 and 19, given a particular HSI scene, spectral-spatial DL-based classifiers have been traditionally trained by extracting randomly selected samples (from the available ground-truth) over the whole image, and cropping spectral-spatial patches of $d \times d \times n_{channels}$ pixel-centered neighbors. In this sense, it is likely that the test set is very close to the train set, or even that part of the test is used in the train set as part of the neighboring region $d \times d$

Table 10
Classification results for the IP, UP and UH datasets considering the CNN models with EMP and Gabor handcrafted features (Ghamisi et al., 2018).

IP dataset				UP dataset				UH dataset			
Class	**CNN**	**EMP CNN**	**GABOR CNN**	**Class**	**CNN**	**EMP CNN**	**GABOR CNN**	**Class**	**CNN**	**EMP CNN**	**GABOR CNN**
Alfalfa	79.25	**85.02**	84.44	Asphalt	88.43	**95.87**	87.75	Grass healthy	82.33	**87.49**	87.47
Corn-notill	90.14	73.45	**91.53**	Meadows	91.64	**99.50**	97.25	Grass stressed	84.30	80.99	**86.01**
Corn-min	98.77	**100.00**	98.77	Gravel	**75.95**	61.12	70.92	Grass synthetic	**95.84**	87.72	78.22
Corn	90.94	92.8	**94.70**	Trees	96.53	94.81	**97.09**	Tree	**92.60**	90.43	85.02
Grass/Pasture	98.85	98.70	**99.28**	Painted metal sheets	98.56	95.15	**98.83**	Soil	99.90	**100.00**	99.89
Grass/Trees	100.00	100.00	100.00	Bare Soil	57.87	**64.84**	64.62	Water	93.00	**97.90**	89.44
Grass/pasture-mowed	95.10	93.13	**95.84**	Bitumen	80.43	**80.63**	76.66	Residential	80.39	**90.48**	90.19
Hay-windrowed	91.20	**92.25**	90.94	Self-Blocking Bricks	98.10	97.26	**99.05**	Commercial	70.42	58.51	**74.44**
Oats	94.34	**94.85**	88.59	Shadows	96.84	96.08	**98.36**	Road	77.77	79.77	**84.42**
Soybeans-notill	100.00	100.00	100.00					Highway	56.08	**64.28**	63.61
Soybeans-min	95.54	99.34	99.34					Railway	75.59	78.37	**80.06**
Soybean-clean	89.66	89.53	89.66					Parking lot1	86.55	78.29	**87.30**
Wheat	100.00	100.00	100.00					Parking lot2	84.21	76.84	**85.06**
Woods	100.00	100.00	97.37					Tennis court	93.11	99.19	**100.00**
Bldg-Grass-Tree-Drives	100.00	100.00	100.00					Running track	**88.37**	77.04	56.95
Stone-steel towers	100.00	100.00	100.00								
OA	91.53	92.40	**92.84**	OA	87.01	91.37	**91.62**	OA	82.75	84.04	**84.12**
AA	95.24	94.94	**95.65**	AA	87.15	87.25	**87.83**	AA	**84.04**	83.33	82.94
K	90.08	91.05	**91.61**	K	83.08	88.67	**89.14**	K	80.61	**82.54**	82.51

Table 11
Overall accuracy (%) achieved by different DL-based approaches when considering different sizes of the input spatial patches. Also, for each model a parameter estimation has been conducted in order to provide an overview of the different architectures.

Spatial Size	SSRN	P-RN	DenseNet	DPN	CapsNet
			IP dataset		
5 × 5	92.83 ± 0.66	**98.80** ± 0.10	97.85 ± 0.28	97.53 ± 0.15	97.79 ± 0.40
7 × 7	97.81 ± 0.34	99.26 ± 0.06	99.24 ± 0.14	99.29 ± 0.06	**99.30** ± 0.11
9 × 9	98.68 ± 0.29	99.64 ± 0.08	99.58 ± 0.09	99.64 ± 0.10	**99.67** ± 0.06
11 × 11	98.70 ± 0.21	**99.82** ± 0.07	99.74 ± 0.08	99.67 ± 0.06	99.74 ± 0.09
			UP dataset		
5 × 5	98.72 ± 0.17	**99.52** ± 0.05	99.13 ± 0.08	99.21 ± 0.11	99.13 ± 0.08
7 × 7	99.54 ± 0.11	**99.81** ± 0.09	99.71 ± 0.10	99.70 ± 0.07	99.75 ± 0.03
9 × 9	99.57 ± 0.54	99.79 ± 0.11	99.73 ± 0.15	**99.88** ± 0.04	99.73 ± 0.10
11 × 11	99.79 ± 0.08	99.92 ± 0.02	99.93 ± 0.03	**99.94** ± 0.03	99.93 ± 0.02
Parameters	360 K.	2.4 M.	1.7 M.	370 K.	9.0 M.

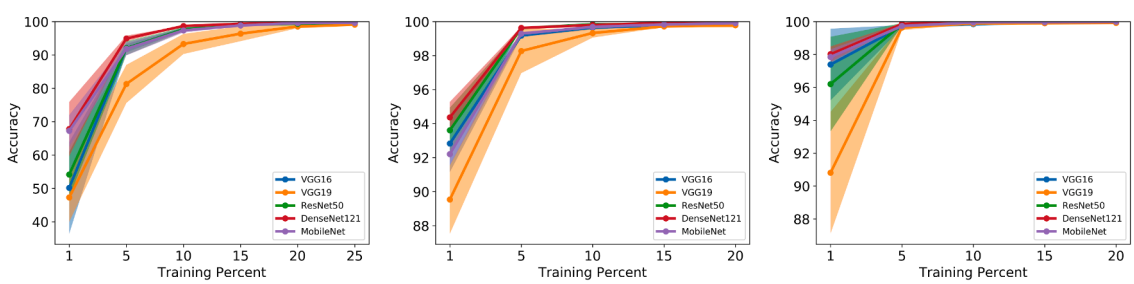
Table 12
Number of samples that the AL-based MLR, CNN1D, CNN2D and CNN3D need to reach a given % of OA for the IP and SV datasets.

Algorithm	Overall Accuracy						
	70%	75%	80%	85%	90%	95%	99%
				IP dataset			
AL-MLR	342	522	–	–	–	–	–
AL-CNN1D	252	352	502	662	–	–	–
AL-CNN2D	222	252	292	352	402	512	662
AL-CNN3D	72	82	112	152	172	232	402
				SV dataset			
AL-MLR	32	32	52	132	412	–	–
AL-CNN1D	32	32	42	62	232	–	–
AL-CNN2D	72	92	122	162	272	412	622
AL-CNN3D	32	32	32	52	72	112	292

selected around the training pixels. Some works (Hänsch et al., 2017) point out that the random sampling strategy has a great influence on the reliability and quality of the obtained solution, because this may significantly facilitate the subsequent classification of the test samples during the inference stage (as they have been previously processed in some way by the network during the training step). As a result, the performance obtained by the model may not be realistic, as artificially optimistic results can be obtained. In order to avoid this important issue, several works (Zhou et al., 2015; Hänsch et al., 2017; Liang et al., 2017; Lange et al., 2018) support the strict spatial-separation between train and test sets, allowing for the acquisition of more realistic accuracy results and a more accurate measurement of the real generalization-power of the model.

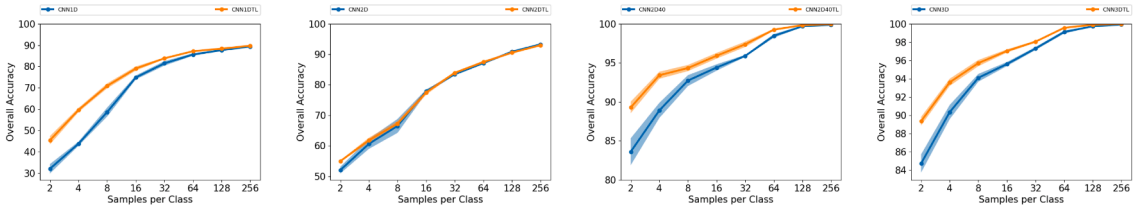
In this context, the aim of this experiment is to compare the results obtained by the spectral (RF, MLR, SVM, MLP, RNN, GRU, LSTM and CNN1D), spatial (CNN2D), and spectral-spatial (CNN2D40 and CNN3D) methods using, on the one hand, the traditional random sampling technique adopted by the methods discussed before in this paper and, on the other hand, a sampling strategy based on selecting spatially separated samples. To pursue this, spatially disjoint training and test sets for the IP and UP datasets (available from the GRSS DASE website (http://dase.grss-ieee.org) have been considered, as depicted on Figs. 18 and 19. For spatial and spectral-spatial methods, neighboring regions of $19 \times 19 \times n_{channels}$ have been cropped from the scenes, setting $n_{channels}$ to 1 and 40 spectral bands for spatial and spectral-spatial methods, respectively, while the spectral-based methods only process the original spectral pixels. The obtained results (in terms of OA) are reported on Table 13. As we can observe, there is a significant performance gap between the obtained results considering randomly selected training samples and spatially disjoint training samples, not only in the spatial and spectral-spatial methods [which is relatively expected due to the aforementioned aspects, as demonstrated by Ham et al. (2005)], but also on purely spectral-based models (which are not really affected by spatial correlations).

Focusing on the IP dataset, it can be noticed that the CNN2D, CNN3D, CNN2D40 and RF are the methods that suffer the most from this phenomenon. While the spectral-spatial methods' performance is significantly affected (reaching OA results in line with those obtained for the UH dataset in the first experiment, whose training and testing samples are spread over a larger area, preventing the possible overlapping effects between the test and train sets), the spectral-based RF is also suffering from a drastic performance drop due to another factor: the suitability of the selected samples. As it was pointed out on Section 2.2, HSI scenes generally suffer from high intraclass variability and interclass similarity, resulting from uncontrolled phenomena such as variations in illumination, presence of areas shaded and/or covered by clouds, and noise distortions, among others. In this sense, the selection of training samples must be carried out very carefully, to avoid situations in which the training and testing samples that belong to the same



(a) IP dataset (b) UP dataset (c) SV dataset

Fig. 16. OA evolution (y-axis) of each considered TL-based classifier with different training percentages (x-axis) over IP, UP and SV datasets. Standard deviation is showed as shaded areas.



(a) TL on CNN1D (b) TL on CNN2D (b) TL on CNN2D40 (c) TL on CNN3D

Fig. 17. Transfer learning experiment from IP to SV datasets, employing (from left to right) the CNN1D, CNN2D, CNND240 and CNN3D models of Table 5.

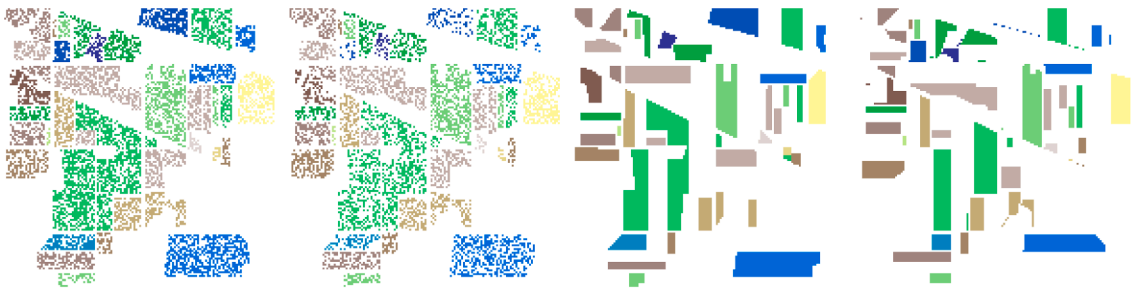
class could be spectrally quite different due to the presence of shadows or noise, for instance. This particularly affects spatial and spectral-spatial convolutional networks (Su et al., 2019).

Focusing on the UP dataset, the same gap between models trained and tested with randomly selected and spatially disjoint samples can be observed. Here, all methods, including the spectral ones (except CNN1D), reduce their OA values in more than 10 percentage points when spatially disjoint training samples are used. In particular, the CNN2D is the most significantly affected method, followed by the RF method. In this sense, it can be concluded that spatial and spectral-spatial methods are significantly affected by the spatial correlation between the training and test sets, which calls for the development of advanced sampling strategies to properly address the high variability of HSI data.

8. Conclusions and future lines

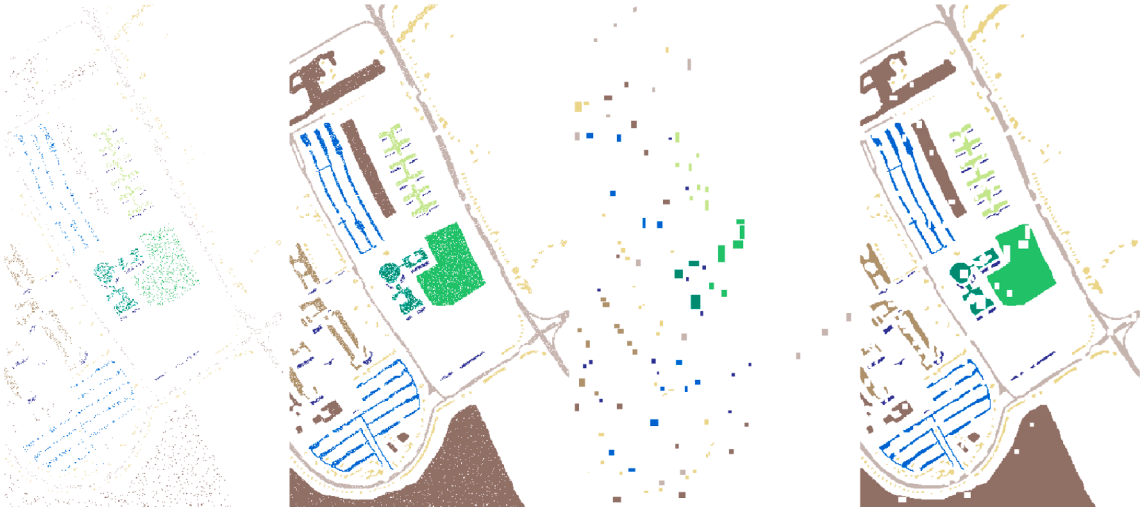
DL methods have revolutionized image analysis and proved to be a powerful tool for processing high-dimensional remotely sensed data, adapting their behavior to the special characteristics of HSI data. In this paper, we have provided an exhaustive review of DL models in the HSI arena. Models based on the CNN architecture have been found to be particularly effective, due to their capacity to extract highly discriminatory features and effectively leverage the spatial-contextual and spectral information contained in HSI data cubes. Traditional and hierarchical structures, composed by chains of blocks concatenated one after another, demonstrate a great generalization power that can be improved through new connections and paths. In fact, the use of standardization techniques, together with the reusability of the information contained in HSI data via residual connections (such as ResNet and DenseNet) and the concatenation of different paths, such as inception modules, have allowed to overcome important problems such as overfitting and the vanishing gradient when few training samples are available, or when very deep structures are implemented. Also, techniques such as AL and TL can help to improve the final performance of very deep neural models in training scenarios dominated by limited training samples, by employing semi-supervised strategies and pre-trained models. In the latter case, additional efforts need to be made in order to perform a more adequate training and adapt the available networks to the special characteristics of HSI data.

One of the main aspects preventing the full adaptation of the discussed paradigms to practical problems is that most of the considered models are highly demanding in computational terms, particularly when applied to complex HSI scenes. However, advances in computer technology and hardware platforms are rapidly allowing to increment the complexity and depth of the networks, making the required fine-tuning processes feasible in a reasonable amount of time. In this sense, there have been several efforts in the field of hardware accelerators that have made possible to implement deep models into embedded processors, GPUs and field programmable gate arrays (FPGAs), which can effectively parallelize the workload of DL-based networks (Randhe et al., 2016; Dong et al., 2017; Zhao et al., 2017b; Haut et al., 2018a). In addition, some research efforts are being carried out to distribute such high computational workloads among various cores using big data strategies, in particular, cloud computing techniques offer great flexibility and scalability, leading to a natural solution for the management of large and complex data HSI datasets. In this regard, we note that more efforts are needed in the remote sensing community in order to deploy cloud computing models, although there are already some works dealing with the exploitation of processing algorithms on cloud architectures (Wu et al., 2016; Haut et al., 2017a; Haut et al., 2017b; Quirita et al., 2017; Haut et al., 2019b). In summary, HPC is an attractive future research direction which can provide efficient mechanisms to address the enormous computational requirements introduced by DL-based HSI data processing, since the acquisition ratios of imaging spectrometers and the volume of future available repositories are expected to be extremely large (Bioucas-Dias et al., 2013), calling for the implementation of complex but faster and more efficient DL-based architectures. Last but not least, another important aspect worth being investigated in future developments is the design of new sample selection methods able to avoid any overlapping between the training and the testing set due to the patch size used by spatial-based methods in the training stage.



(a) Random Train (b) Random Test (c) Disjoint Train (d) Disjoint Test

Fig. 18. Comparison between the random selection method and the selection of spatially disjoint samples on the IP dataset, considering the same number of samples per class.



(a) Random Train (b) Random Test (c) Disjoint Train (d) Disjoint Test

Fig. 19. Comparison between the random selection method and the selection of spatially disjoint samples on the UP scene, considering the same number of samples per class.

Table 13
Comparison (in terms of OA) between different HSI classification models trained using randomly selected samples and spatial-disjoint samples during the training and inference stages.

	INDIAN PINES			PAVIA UNIVERSITY		
	Disjoint	Random	Diff	Disjoint	Random	Diff
RF	65.79	80.31	14.52	69.64	85.81	16.17
MLR	78.22	83.15	4.93	72.23	86.75	14.52
SVM	85.08	90.56	5.48	77.80	92.36	14.56
MLP	83.81	90.61	6.8	81.96	93.06	11.1
RNN	79.40	86.98	7.58	76.77	91.19	14,42
GRU	83.21	90.28	7.07	81.47	92.53	11.06
LSTM	82.94	90.76	7.82	79.54	91.09	11.55
CNN1D	84.94	91.96	7.02	87.59	94.15	6.56
CNN2D	56.66	99.77	43.11	76.47	98.22	21.75
CNN2D40	82.97	99.99	17.02	84.98	99.94	14.96
CNN3D	79.58	99.96	20.38	86.82	99.95	13.13

Acknowledgements

This work has been suported by:

- Spanish 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.
- 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).
- European Union's Horizon 2020 research and innovation programme under grant agreement No. 734541 (EOXPOSURE).

The authors would like to gratefully thank the Associate Editor and the two Anonymous Reviewers for their outstanding comments and suggestions, which greatly helped us to improve the technical quality and presentation of the manuscript.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., et al., 2016a. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467.

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al., 2016b. Tensorflow: A system for large-scale machine learning. In: 12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16). pp. 265–283.

Abbate, G., Fiumi, L., Lorenzo, C.D., Vintila, R., May 2003. Evaluation of remote sensing data for urban planning. applicative examples by means of multispectral and hyperspectral data. In: 2003 2nd GRSS/ISPRS Joint Workshop on Remote Sensing and Data Fusion over Urban Areas. pp. 201–205.

Ablin, R., Sulochana, C.H., 2013. A survey of hyperspectral image classification in remote sensing. Int. J. Adv. Res. Comput. Commun. Eng. 2 (8), 2986–3000.

Acosta, I.C.C., Khodadadzadeh, M., Tusa, L., Ghamisi, P., Gloaguen, R., 2019. A machine learning framework for drill-core mineral mapping using hyperspectral and high-resolution mineralogical data fusion. IEEE J. Sel. Top. Appl. Earth Obser. Remote Sens.

Acquarelli, J., Marchiori, E., Buydens, L., Tran, T., Laarhoven, T., 2018. Spectral-spatial classification of hyperspectral images: Three tricks and a new learning setting. Remote Sensing 10 (7), 1156.

Agostinelli, F., Hoffman, M.D., Sadowski, P.J., Baldi, P., 2014. Learning activation functions to improve deep neural networks. arXiv preprint arXiv:1412.6830. http://arxiv.org/abs/1412.6830.

Ahmad, M., Protasov, S., Khan, A.M., 2017. Hyperspectral band selection using unsupervised non-linear deep auto encoder to train external classifiers. CoRR abs/1705.06920. URL http://arxiv.org/abs/1705.06920.

Al-khafaji, S.L., Zhou, J., Zia, A., Liew, A.W., 2018. Spectral-spatial scale invariant feature transform for hyperspectral images. IEEE Trans. Image Process. 27 (2), 837–850.

Anand, R., Veni, S., Aravinth, J., 2017. Big data challenges in airborne hyperspectral image for urban landuse classification. In: 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 1808–1814.

Anwer, R.M., Khan, F.S., van de Weijer, J., Molinier, M., Laaksonen, J., 2018. Binary patterns encoded convolutional neural networks for texture recognition and remote sensing scene classification. ISPRS J. Photogramm. Remote Sensing 138, 74–85.

Aptoula, E., Ozdemir, M.C., Yanikoglu, B., 2016. Deep learning with attribute profiles for hyperspectral image classification. IEEE Geosci. Remote Sens. Lett. 13 (12), 1970–1974.

Ardouin, J.P., Levesque, J., Rea, T.A., 2007. A demonstration of hyperspectral image exploitation for military applications. In: 2007 10th International Conference on Information Fusion, pp. 1–8.

Aslett, Z., Taranik, J.V., Riley, D.N., 2018. Mapping rock forming minerals at boundary canyon, death valey national park, california, using aerial sebass thermal infrared hyperspectral image data. Int. J. Appl. Earth Obs. Geoinf. 64, 326–339.

Audebert, N., Le Saux, B., Lefèvre, S., 2019. Deep learning for classification of hyperspectral data: A comparative review. IEEE Geosci. Remote Sens. Mag. 7 (2), 159–173.

Ba, J., Frey, B., 2013. Adaptive dropout for training deep neural networks. In: Burges, C.J.C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q. (Eds.), Advances in Neural Information Processing Systems 26. Curran Associates, Inc., pp. 3084–3092.

Ba, J.L., Kiros, J.R., Hinton, G.E., 2016. Layer normalization. arXiv preprint arXiv:1607.06450.

Babey, S., Anger, C., 1989. A compact airborne spectrographic imager (casi). In: Quantitative Remote Sensing: An Economic Tool for the Nineties, vol. 1. pp. 1028–1031.

Bach, F., 2017. Breaking the curse of dimensionality with convex neural networks. J. Machine Learn. Res. 18 (19), 1–53.
Baldi, P., Hornik, K., 1989. Neural networks and principal component analysis: Learning from examples without local minima. Neural Networks 2 (1), 53–58.
Ball, J.E., Anderson, D.T., Chan, C.S., 2017. Comprehensive survey of deep learning in remote sensing: theories, tools, and challenges for the community. J. Appl. Remote Sens. 11, 11–54.
Bannari, A., Pacheco, A., Staenz, K., McNairn, H., Omari, K., 2006. Estimating and mapping crop residues cover on agricultural lands using hyperspectral and ikonos data. Remote Sens. Environ. 104 (4), 447–459.
Bellman, R., 2015. Adaptive Control Processes: A Guided Tour. Princeton Legacy Library. Princeton University Press.
Benediktsson, J.A., Sveinsson, J.R., 2003. Multisource remote sensing data classification based on consensus and pruning. IEEE Trans. Geosci. Remote Sens. 41 (4), 932–936.
Benediktsson, J.A., Swain, P.H., Ersoy, O.K., 1993. Conjugate-gradient neural networks in classification of multisource and very-high-dimensional remote sensing data. Int. J. Remote Sens. 14 (15), 2883–2903.
Bengio, Y., 2009. Learning deep architectures for ai. Found. Trends Machine Learn. 2 (1), 1–127.
Bengio, Y., Courville, A., Vincent, P., 2013. Representation learning: A review and new perspectives. IEEE Trans. Pattern Anal. Machine Intell. 35 (8), 1798–1828.
Bengio, Y., Courville, A.C., Vincent, P., 2012. Unsupervised feature learning and deep learning: A review and new perspectives. CoRR, abs/1206.5538 1, 2012.
Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H., 2007a. Greedy layer-wise training of deep networks. In: Schölkopf, B., Platt, J.C., Hoffman, T. (Eds.), Advances in Neural Information Processing Systems 19. MIT Press, pp. 153–160.
Bengio, Y., LeCun, Y., et al., 2007b. Scaling learning algorithms towards ai. Large-scale Kernel Machines 34 (5), 1–41.
Benítez, J.M., Castro, J.L., Requena, I., 1997. Are artificial neural networks black boxes? IEEE Trans. Neural Networks 8 (5), 1156–1164.
Bhardwaj, K., Patra, S., 2018. An unsupervised technique for optimal feature selection in attribute profiles for spectral-spatial classification of hyperspectral images. ISPRS J. Photogramm. Remote Sens. 138, 139–150.
Bioucas-Dias, J.M., Plaza, A., Camps-Valls, G., Scheunders, P., Nasrabadi, N., Chanussot, J., 2013. Hyperspectral remote sensing data analysis and future challenges. IEEE Geosci. Remote Sens. Mag. 1 (2), 6–36.
Bioucas-Dias, J.M., Plaza, A., Dobigeon, N., Parente, M., Du, Q., Gader, P., Chanussot, J., 2012. Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 5 (2), 354–379.
Bishop, C., 1995. Neural Networks for Pattern Recognition. Advanced Texts in Econometrics. Clarendon Press.
Bjorck, N., Gomes, C.P., Selman, B., Weinberger, K.Q., 2018. Understanding batch normalization. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 31. Curran Associates, Inc., pp. 7694–7705.
Blum, A., Rivest, R.L., 1989. Training a 3-node neural network is np-complete. In: Advances in Neural Information Processing Systems. pp. 494–501.
Boureau, Y.-L., Ponce, J., LeCun, Y., 2010. A theoretical analysis of feature pooling in visual recognition. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10). pp. 111–118.
Brendel, W., Bethge, M., 2019. Approximating CNNs with bag-of-local-features models works surprisingly well on imagenet. In: International Conference on Learning Representations, pp. 15.
Briottet, X., Boucher, Y., Dimmeler, A., Malaplate, A., Cini, A., Diani, M., Bekman, H., Schwering, P., Skauli, T., Kasen, I., et al., 2006. Military applications of hyperspectral imagery. In: Targets and backgrounds XII: Characterization and representation. Vol. 6239. International Society for Optics and Photonics, p. 62390B.
Bruce, L.M., Koger, C.H., Li, J., 2002. Dimensionality reduction of hyperspectral data using discrete wavelet transform feature extraction. IEEE Trans. Geosci. Remote Sensing 40 (10), 2331–2338.
Büchel, J., Ersoy, O., 2018. Ladder networks for semi-supervised hyperspectral image classification. arXiv preprint arXiv:1812.01222.
Bue, B.D., Thompson, D.R., Eastwood, M., Green, R.O., Gao, B.C., Keymeulen, D., Sarture, C.M., Mazer, A.S., Luong, H.H., 2015. Real-time atmospheric correction of aviris-ng imagery. IEEE Trans. Geosci. Remote Sens. 53 (12), 6419–6428.
Calin, M.A., Parasca, S.V., Manea, D., 2018. Comparison of spectral angle mapper and support vector machine classification methods for mapping skin burn using hyperspectral imaging. In: Unconventional Optical Imaging. Vol. 10677. International Society for Optics and Photonics, p. 106773P.
Camps-Valls, G., 2016. Kernel spectral angle mapper. Electron. Lett. 52 (14), 1218–1220.
Camps-Valls, G., Gomez-Chova, L., Muñoz-Marí, J., Vila-Francés, J., Calpe-Maravilla, J., 2006. Composite kernels for hyperspectral image classification. IEEE Geosci. Remote Sens. Letters 3 (1), 93–97.
Camps-Valls, G., Tuia, D., Bruzzone, L., Benediktsson, J.A., 2014. Advances in hyperspectral image classification: Earth monitoring with statistical learning methods. IEEE Signal Process. Mag. 31 (1), 45–54.
Cao, X., Zhou, F., Xu, L., Meng, D., Xu, Z., Paisley, J., 2018. Hyperspectral image classification with markov random fields and a convolutional neural network. IEEE Trans. Image Process. 27 (5), 2354–2367.
Cariou, C., Chehdi, K., 2015. Unsupervised nearest neighbors clustering with application to hyperspectral images. IEEE J. Sel. Top. Signal Process. 9 (6), 1105–1116.
Caruana, R., Lawrence, S., Giles, C.L., 2001. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In: Advances in Neural Information Processing Systems. pp. 402–408.
Chabrillat, S., Milewski, R., Schmid, T., Rodriguez, M., Escribano, P., Pelayo, M., Palacios-Orueta, A., July 2014. Potential of hyperspectral imagery for the spatial assessment of soil erosion stages in agricultural semi-arid spain at different scales. In: 2014 IEEE Geoscience and Remote Sensing Symposium. pp. 2918–2921.
Chang, C.-I., 2007. Hyperspectral Data Exploitation: Theory and Applications. John Wiley & Sons.
Charles, A.S., Olshausen, B.A., Rozell, C.J., 2011. Learning sparse codes for hyperspectral imagery. IEEE J. Sel. Top. Signal Process. 5 (5), 963–978.
Charmisha, K., Sowmya, V., Soman, K., 2018. Dimensionally reduced features for hyperspectral image classification using deep learning. In: International Conference on Communications and Cyber Physical Engineering 2018. Springer, pp. 171–179.
Chen, G., Qian, S.-E., 2011. Denoising of hyperspectral imagery using principal component analysis and wavelet shrinkage. IEEE Trans. Geosci. Remote Sens. 49 (3), 973–980.
Chen, S., Wang, Y., 2014. Convolutional neural network and convex optimization. Dept. of Elect. and Comput. Eng., Univ. of California at San Diego, San Diego, CA, USA, Tech. Rep.
Chen, X., Xiang, S., Liu, C.-L., Pan, C.-H., 2014a. Vehicle Detection in Satellite Images by Hybrid Deep Convolutional Neural Networks. IEEE Geosci. Remote Sens. Lett. 11 (10), 1797–1801.
Chen, Y., Jiang, H., Li, C., Jia, X., Ghamisi, P., 2016. Deep feature extraction and classification of hyperspectral images based on convolutional neural networks. IEEE Trans. Geosci. Remote Sens. 54 (10), 6232–6251.
Chen, Y., Li, J., Xiao, H., Jin, X., Yan, S., Feng, J., 2017a. Dual path networks. In: Advances in Neural Information Processing Systems. pp. 4467–4475.
Chen, Y., Lin, Z., Zhao, X., Wang, G., Gu, Y., 2014b. Deep Learning-Based Classification of Hyperspectral Data. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 7 (6), 2094–2107.
Chen, Y., Wang, Y., Gu, Y., He, X., Ghamisi, P., Jia, X., 2019a. Deep learning ensemble for hyperspectral image classification. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.
Chen, Y., Zhao, X., Jia, X., 2015. Spectral-Spatial Classification of Hyperspectral Data Based on Deep Belief Network. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 8 (6), 2381–2392.
Chen, Y., Zhu, K., Zhu, L., He, X., Ghamisi, P., Benediktsson, J.A., 2019b. Automatic design of convolutional neural network for hyperspectral image classification. IEEE Trans. Geosci. Remote Sens.
Chen, Y., Zhu, L., Ghamisi, P., Jia, X., Li, G., Tang, L., 2017b. Hyperspectral images classification with gabor filtering and convolutional neural network. IEEE Geosci. Remote Sens. Lett. 14 (12), 2355–2359.
Cheng, G., Han, J., Lu, X., 2017a. Remote sensing image scene classification: Benchmark and state of the art. Proc. IEEE 105 (10), 1865–1883.
Cheng, Y., Wang, D., Zhou, P., Zhang, T., 2017b. A survey of model compression and acceleration for deep neural networks. arXiv preprint arXiv:1710.09282.
Cho, K., Van Merriënboer, B., Bahdanau, D., Bengio, Y., 2014. On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259.
Chollet, F., 2017. Xception: Deep learning with depthwise separable convolutions. arXiv preprint, 1610–02357.
Chutia, D., Bhattacharyya, D., Sarma, K.K., Kalita, R., Sudhakar, S., 2016. Hyperspectral remote sensing classifications: a perspective survey. Trans. GIS 20 (4), 463–490.
Cocks, T., Jenssen, R., Stewart, A., Wilson, I., Shields, T., 1998. The hymaptm airborne hyperspectral sensor: the system, calibration and performance. In: Proceedings of the 1st EARSeL workshop on Imaging Spectroscopy. EARSeL, pp. 37–42.
Collobert, R., Bengio, S., 2004. Links between perceptrons, mlps and svms. In: Proceedings of the Twenty-first International Conference on Machine Learning. ACM, pp. 23.
Collobert, R., Bengio, S., Mariéthoz, J., 2002. Torch: a modular machine learning software library. Tech. Rep., Idiap.
Coops, N.C., Smith, M.L., Martin, M.E., Ollinger, S.V., 2003. Prediction of eucalypt foliage nitrogen content from satellite-derived hyperspectral data. IEEE Trans. Geosci. Remote Sens. 41 (6), 1338–1346.
Cybenko, G., 1989. Approximation by superpositions of a sigmoidal function. Math. Control, Signals Syst. 2 (4), 303–314.
Dalla Mura, M., Villa, A., Benediktsson, J.A., Chanussot, J., Bruzzone, L., 2011. Classification of hyperspectral images by using extended morphological attribute profiles and independent component analysis. IEEE Geosci. Remote Sens. Lett. 8 (3), 542–546.
Dauphin, Y.N., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., Bengio, Y., 2014. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In: Advances in Neural Information Processing Systems. pp. 2933–2941.
Debes, C., Merentitis, A., Heremans, R., Hahn, J., Frangiadakis, N., van Kasteren, T., Liao, W., Bellens, R., Pižurica, A., Gautama, S., Philips, W., Prasad, S., Du, Q., Pacifici, F., 2014. Hyperspectral and lidar data fusion: Outcome of the 2013 grss data fusion contest. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 7 (6), 2405–2418.
Defferrard, M., Bresson, X., Vandergheynst, P., 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In: Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 29. Curran Associates, Inc., pp. 3844–3852.
Deng, C., Xue, Y., Liu, X., Li, C., Tao, D., 2019. Active transfer learning network: A unified deep joint spectral–spatial feature learning model for hyperspectral image classification. IEEE Trans. Geosci. Remote Sens. 57 (3), 1741–1754.
Deng, F., Pu, S., Chen, X., Shi, Y., Yuan, T., Pu, S., 2018. Hyperspectral image classification with capsule network using limited training samples. Sensors 18 (9).
Ding, C., Li, Y., Xia, Y., Wei, W., Zhang, L., Zhang, Y., 2017. Convolutional neural networks based hyperspectral image classification method with adaptive kernels. Remote Sens. 9 (6), 618.
Dong, H., Li, T., Leng, J., Kong, L., Bai, G., 2017. Gcn: Gpu-based cube cnn framework for

hyperspectral image classification. In: 2017 46th International Conference on Parallel Processing (ICPP), pp. 41–49.
Dong, H., Zhang, L., Zou, B., 2019. Band attention convolutional networks for hyperspectral image classification. arXiv preprint arXiv:1906.04379.
Du, J., Li, Z., 2018. A hyperspectral target detection framework with subtraction pixel pair features. IEEE Access 6, 45562–45577.
Du, Q., Chang, C.-I., 2001. A linear constrained distance-based discriminant analysis for hyperspectral image classification. Pattern Recogn. 34 (2), 361–373.
Duchi, J., Hazan, E., Singer, Y., 2011. Adaptive subgradient methods for online learning and stochastic optimization. J. Machine Learn. Res. 12 (Jul), 2121–2159.
Dugas, C., Bengio, Y., Bélisle, F., Nadeau, C., Garcia, R., 2001. Incorporating second-order functional knowledge for better option pricing. In: Advances in Neural Information Processing Systems. pp. 472–478.
Dumke, I., Nornes, S.M., Purser, A., Marcon, Y., Ludvigsen, M., Ellefmo, S.L., Johnsen, G., Søreide, F., 2018. First hyperspectral imaging survey of the deep seafloor: High-resolution mapping of manganese nodules. Remote Sens. Environ. 209, 19–30.
Eckardt, A., Horack, J., Lehmann, F., Krutz, D., Drescher, J., Whorton, M., Soutullo, M., 2015. Desis (dlr earth sensing imaging spectrometer for the iss-muses platform). In: 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS). IEEE, pp. 1457–1459.
Eismann, M.T., Hardie, R.C., 2005. Hyperspectral resolution enhancement using high-resolution multispectral imagery with arbitrary response functions. IEEE Trans. Geosci. Remote Sens. 43 (3), 455–465.
El-Magd, I.A., El-Zeiny, A., 2014. Quantitative hyperspectral analysis for characterization of the coastal water from damietta to port said, egypt. Egyp. J. Remote Sens. Space Sci. 17 (1), 61–76.
El-Sharkawy, Y.H., Elbasuney, S., 2019. Hyperspectral imaging: Anew prospective for remote recognition of explosive materials. Remote Sensing Appl.: Soc. Environ. 13, 31–38.
Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., Vincent, P., Bengio, S., 2010. Why does unsupervised pre-training help deep learning? J. Machine Learn. Res. 11 (Feb), 625–660.
Fang, B., Li, Y., Zhang, H., Chan, J., 2018. Semi-supervised deep learning classification for hyperspectral image based on dual-strategy sample selection. Remote Sens. 10 (4), 574.
Fang, B., Li, Y., Zhang, H., Chan, J.C.-W., 2019. Hyperspectral images classification based on dense convolutional networks with spectral-wise attention mechanism. Remote Sens. 11 (2), 159.
Fauvel, M., Benediktsson, J.A., Chanussot, J., Sveinsson, J.R., 2008. Spectral and spatial classification of hyperspectral data using svms and morphological profiles. IEEE Trans. Geosci. Remote Sens. 46 (11), 3804–3814.
Fauvel, M., Tarabalka, Y., Benediktsson, J.A., Chanussot, J., Tilton, J.C., 2013. Advances in spectral-spatial classification of hyperspectral images. Proc. IEEE 101 (3), 652–675.
Feingersh, T., Dor, E.B., 2015. Shalom–a commercial hyperspectral space mission. Opt. Payloads Space Missions 247–263.
Feng, W., Qi, S., Heng, Y., Zhou, Y., Wu, Y., Liu, W., He, L., Li, X., 2017. Canopy vegetation indices from in situ hyperspectral data to assess plant water status of winter wheat under powdery mildew stress. Front. Plant Sci. 8 (1219).
Fernandez, D., Gonzalez, C., Mozos, D., Lopez, S., 2016. Fpga implementation of the principal component analysis algorithm for dimensionality reduction of hyperspectral images. J. Real-Time Image Proc. 1–12.
Fey, M., Lenssen, J.E., 2019. Fast graph representation learning with pytorch geometric. arXiv preprint arXiv:1903.02428.
Field, D.J., 1999. Wavelets, vision and the statistics of natural scenes. Philosoph. Trans. Roy. Soc. London A: Math., Phys. Eng. Sci. 357 (1760), 2527–2542.
Fisher, P., 1997. The pixel: a snare and a delusion. Int. J. Remote Sens. 18 (3), 679–685.
Galeazzi, C., Sacchetti, A., Cisbani, A., Babini, G., 2008. The prisma program. In: Geoscience and Remote Sensing Symposium, 2008. IGARSS 2008. IEEE International, vol. 4. IEEE, pp. IV–105.
Gao, H., Lin, S., Yang, Y., Li, C., Yang, M., 2018a. Convolution neural network based on two-dimensional spectrum for hyperspectral image classification. J. Sensors 2018, 13.
Gao, Q., Lim, S., Jia, X., 2018b. Hyperspectral image classification using convolutional neural networks and multiple feature learning. Remote Sens. 10 (2), 299.
Ghamisi, P., Chen, Y., Zhu, X.X., 2016. A self-improving convolution neural network for the classification of hyperspectral data. IEEE Geosci. Remote Sens. Lett. 13 (10), 1537–1541.
Ghamisi, P., Maggiori, E., Li, S., Souza, R., Tarablaka, Y., Moser, G., De Giorgi, A., Fang, L., Chen, Y., Chi, M., et al., 2018. New frontiers in spectral-spatial hyperspectral image classification: The latest advances based on mathematical morphology, markov random fields, segmentation, sparse representation, and deep learning. IEEE Geosci. Remote Sens. Mag. 6 (3), 10–43.
Ghamisi, P., Plaza, J., Chen, Y., Li, J., Plaza, A.J., 2017a. Advanced spectral classifiers for hyperspectral images: A review. IEEE Geosci. Remote Sens. Mag. 5 (1), 8–32.
Ghamisi, P., Yokoya, N., Li, J., Liao, W., Liu, S., Plaza, J., Rasti, B., Plaza, A., 2017b. Advances in hyperspectral image and signal processing: A comprehensive overview of the state of the art. IEEE Geosci. Remote Sens. Mag. 5 (4), 37–78.
Ghiasi, G., Lin, T.-Y., Le, Q.V., 2018. Dropblock: A regularization method for convolutional networks. In: Advances in Neural Information Processing Systems. pp. 10750–10760.
Gitman, I., Ginsburg, B., 2017. Comparison of batch normalization and weight normalization algorithms for the large-scale image classification. arXiv preprint arXiv:1709.08145.
Glorot, X., Bengio, Y., 13–15 May 2010. Understanding the difficulty of training deep feedforward neural networks. In: Teh, Y.W., Titterington, M. (Eds.), Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. Vol. 9 of Proceedings of Machine Learning Research. PMLR, Chia Laguna Resort, Sardinia, Italy, pp. 249–256.
Goetz, A.F.H., Vane, G., Solomon, J.E., Rock, B.N., 1985. Imaging Spectrometry for Earth Remote Sensing. Science 228 (4704), 1147–1153.
Gomez, C., Drost, A., Roger, J.-M., 2015. Analysis of the uncertainties affecting predictions of clay contents from vnir/swir hyperspectral data. Remote Sens. Environ. 156, 58–70.
Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.
Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., 2014. Generative adversarial nets. In: Advances in Neural Information Processing Systems. pp. 2672–2680.
Green, A.A., Berman, M., Switzer, P., Craig, M.D., 1988. A transformation for ordering multispectral data in terms of image quality with implications for noise removal. IEEE Trans. Geosci. Remote Sens. 26 (1), 65–74.
Green, R.O., Eastwood, M.L., Sarture, C.M., Chrien, T.G., Aronsson, M., Chippendale, B.J., Faust, J.A., Pavri, B.E., Chovit, C.J., Solis, M., Olah, M.R., Williams, O., 1998. Imaging spectroscopy and the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS). Remote Sens. Environ. 65 (3), 227–248.
Große-Stoltenberg, A., Hellmann, C., Werner, C., Oldeland, J., Thiele, J., 2016. Evaluation of continuous vnir-swir spectra versus narrowband hyperspectral indices to discriminate the invasive acacia longifolia within a mediterranean dune ecosystem. Remote Sens. 8 (4).
Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., Wang, G., Cai, J., et al., 2018. Recent advances in convolutional neural networks. Pattern Recogn. 77, 354–377.
Guanter, L., Kaufmann, H., Segl, K., Foerster, S., Rogass, C., Chabrillat, S., Kuester, T., Hollstein, A., Rossner, G., Chlebek, C., et al., 2015. The enmap spaceborne imaging spectroscopy mission for earth observation. Remote Sens. 7 (7), 8830–8857.
Guo, A.J., Zhu, F., 2018. A cnn-based spatial feature fusion algorithm for hyperspectral imagery classification. arXiv preprint arXiv:1801.10355.
Guo, Y., Han, S., Cao, H., Zhang, Y., Wang, Q., 2018. Guided filter based deep recurrent neural networks for hyperspectral image classification. Procedia Computer Science 129, 219–223, 2017 International Conference on Identification, Information and Knowledge in the Internet of Things.
Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S., Lew, M.S., 2016. Deep learning for visual understanding: A review. Neurocomputing 187, 27–48 recent Developments on Deep Big Vision.
Guofeng, T., Yong, L., Lihao, C., Chen, J., June 2017. A dbn for hyperspectral remote sensing image classification. In: 2017 12th IEEE Conference on Industrial Electronics and Applications (ICIEA). pp. 1757–1762.
Haboudane, D., Miller, J.R., Pattey, E., Zarco-Tejada, P.J., Strachan, I.B., 2004. Hyperspectral vegetation indices and novel algorithms for predicting green lai of crop canopies: Modeling and validation in the context of precision agriculture. Remote Sens. Environ. 90 (3), 337–352.
Ham, J., Chen, Y., Crawford, M.M., Ghosh, J., 2005. Investigation of the random forest framework for classification of hyperspectral data. IEEE Trans. Geosci. Remote Sens. 43 (3), 492–501.
Han, Y., Li, J., Zhang, Y., Hong, Z., Wang, J., 2017. Sea ice detection based on an improved similarity measurement method using hyperspectral data. Sensors 17 (5), 1124.
Hänsch, R., Ley, A., Hellwich, O., 2017. Correct and still wrong: The relationship between sampling strategies and the estimation of the generalization error. In: 2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS). IEEE, pp. 3672–3675.
Hao, S., Wang, W., Ye, Y., Nie, T., Bruzzone, L., 2018. Two-stream deep architecture for hyperspectral image classification. IEEE Trans. Geosci. Remote Sens. 56 (4), 2349–2361.
Hassanzadeh, A., Kaarna, A., Kauranne, T., 2017. Unsupervised multi-manifold classification of hyperspectral remote sensing images with contractive autoencoder. In: Sharma, P., Bianchi, F.M. (Eds.), Image Analysis. Springer International Publishing, Cham, pp. 169–180.
Hassanzadeh, A., Kaarna, A., Kauranne, T., 2018. Sequential spectral clustering of hyperspectral remote sensing image over bipartite graph. Appl. Soft Comput. 73, 727–734.
Haut, J., Paoletti, M., Paz-Gallardo, A., Plaza, J., Plaza, A., 2017a. Cloud implementation of logistic regression for hyperspectral image classification. In: Vigo-Aguiar, J. (Ed.), Proceedings of the 17th International Conference on Computational and Mathematical Methods in Science and Engineering, CMMSE 2017. Costa Ballena (Rota), Cádiz, Spain, pp. 1063–2321.
Haut, J., Paoletti, M., Plaza, J., Plaza, A., 2017b. Cloud implementation of the K-means algorithm for hyperspectral image analysis. J. Supercomput. 73 (1).
Haut, J., Paoletti, M., Plaza, J., Plaza, A., 2019a. Hyperspectral image classification using random occlusion data augmentation. IEEE Geosci. Remote Sens. Lett.
Haut, J.M., Bernabé, S., Paoletti, M.E., Fernandez-Beltran, R., Plaza, A., Plaza, J., 2018a. Low–high-power consumption architectures for deep-learning models applied to hyperspectral image classification. IEEE Geosci. Remote Sens. Lett. 16 (5), 776–780.
Haut, J.M., Fernandez-Beltran, R., Paoletti, M.E., Plaza, J., Plaza, A., Pla, F., 2018b. A new deep generative network for unsupervised remote sensing single-image super-resolution. IEEE Trans. Geosci. Remote Sens. 1–19.
Haut, J.M., Gallardo, J.A., Paoletti, M.E., Cavallaro, G., Plaza, J., Plaza, A., Riedel, M., 2019b. Cloud deep networks for hyperspectral image analysis. IEEE Trans. Geosci. Remote Sens.
Haut, J.M., Paoletti, M.E., Plaza, J., Li, J., Plaza, A., 2018c. Active learning with convolutional neural networks for hyperspectral image classification using a new bayesian approach. IEEE Trans. Geosci. Remote Sens. 1–22.
Haut, J.M., Paoletti, M.E., Plaza, J., Plaza, A., 2018d. Fast dimensionality reduction and

classification of hyperspectral images with extreme learning machines. J. Real-Time Image Proc. 1–24.
Haut, J.M., Paoletti, M.E., Plaza, J., Plaza, A., Li, J., 2019. Visual attention-driven hyperspectral image classification. IEEE Trans. Geosci. Remote Sens. 1–16.
He, K., Sun, J., Tang, X., 2013. Guided image filtering. IEEE Trans. Pattern Anal. Machine Intell. 35 (6), 1397–1409.
He, K., Zhang, X., Ren, S., Sun, J., 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: 2015 IEEE International Conference on Computer Vision (ICCV), pp. 1026–1034.
He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778.
He, L., Li, J., Plaza, A., Li, Y., 2017a. Discriminative low-rank gabor filtering for spectral-spatial hyperspectral image classification. IEEE Trans. Geosci. Remote Sens. 55 (3), 1381–1395.
He, N., Paoletti, M.E., Haut, J.n.M., Fang, L., Li, S., Plaza, A., Plaza, J., 2018. Feature extraction with multiscale covariance maps for hyperspectral image classification. IEEE Trans. Geosci. Remote Sens. 1–15.
He, X., Chen, Y., 2019. Optimized input for cnn-based hyperspectral image classification using spatial transformer network. IEEE Geosci. Remote Sens. Lett.
He, Z., Liu, H., Wang, Y., Hu, J., 2017b. Generative adversarial networks-based semi-supervised learning for hyperspectral image classification. Remote Sens. 9 (10), 1042.
Heldens, W., Heiden, U., Esch, T., Stein, E., Müller, A., 2011. Can the future enmap mission contribute to urban applications? a literature survey. Remote Sens. 3 (9), 1817–1846.
Heylen, R., Parente, M., Gader, P., 2014. A review of nonlinear hyperspectral unmixing methods. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 7 (6), 1844–1868.
Hinton, G., Salakhutdinov, R., 2006. Reducing the dimensionality of data with neural networks. Science 313, 504–507.
Hinton, G.E., Osindero, S., Teh, Y.-W., 2006. A fast learning algorithm for deep belief nets. Neural Comput. 18 (7), 1527–1554.
Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R., 2012. Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580.
Hinton, G.E., Zemel, R.S., 1993. Autoencoders, minimum description length and helmholtz free energy. In: Proceedings of the 6th International Conference on Neural Information Processing Systems. NIPS'93. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 3–10.
Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9 (8), 1735–1780.
Hornik, K., 1991. Approximation capabilities of multilayer feedforward networks. Neural Networks 4 (2), 251–257.
Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H., 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.
Hu, W., Huang, Y., Wei, L., Zhang, F., Li, H., 2015. Deep convolutional neural networks for hyperspectral image classification. J. Sensors.
Huadong, G., Jianmin, X., Guoqiang, N., Jialing, M., 2001. A new airborne earth observing system and its applications. In: IGARSS 2001. Scanning the Present and Resolving the Future. Proceedings. IEEE 2001 International Geoscience and Remote Sensing Symposium (Cat. No. 01CH37217). Vol. 1. pp. 549–551 vol 1.
Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q., 2017. Densely connected convolutional networks. In: CVPR. Vol. 1. p. 3.
Huang, X., Belongie, S., 2017. Arbitrary style transfer in real-time with adaptive instance normalization. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1501–1510.
Huang, X., Zhang, L., 2013. An svm ensemble approach combining spectral, structural, and semantic features for the classification of high-resolution remotely sensed imagery. IEEE Trans. Geosci. Remote Sens. 51 (1), 257–272.
Huangi, L., Huangi, L., Yang, D., Lang, B., Deng, J., 2018. Decorrelated batch normalization. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 791–800.
Hughes, G., 1968. On the mean accuracy of statistical pattern recognizers. IEEE Trans. Inf. Theory 14 (1), 55–63.
Ioffe, S., 2017. Batch renormalization: Towards reducing minibatch dependence in batch-normalized models. In: Advances in Neural Information Processing Systems. pp. 1945–1953.
Ioffe, S., Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167.
Iyer, R.P., Raveendran, A., Bhuvana, S.K.T., Kavitha, R., 2017. Hyperspectral image analysis techniques on remote sensing. In: 2017 Third International Conference on Sensing, Signal Processing and Security (ICSSS), pp. 392–396.
Jia, P., Zhang, M., Yu, W., Shen, F., Shen, Y., 2016. Convolutional neural network based classification for hyperspectral data. In: 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 5075–5078.
Jiao, L., Liang, M., Chen, H., Yang, S., Liu, H., Cao, X., 2017. Deep fully convolutional network-based spatial distribution prediction for hyperspectral image classification. IEEE Trans. Geosci. Remote Sens. 55 (10), 5585–5599.
Jiménez, L.O., Rivera-Medina, J.L., Rodríguez-Díaz, E., Arzuaga-Cruz, E., Ramírez-Vélez, M., 2005. Integration of spatial and spectral information by means of unsupervised extraction and classification for homogenous objects applied to multispectral and hyperspectral data. IEEE Trans. Geosci. Remote Sens. 43 (4), 844–851.
Jing, L., Tian, Y., 2019. Self-supervised visual feature learning with deep neural networks: A survey. arXiv preprint arXiv:1902.06162.
Jolliffe, I., 2002. Principal Component Analysis. Springer Series in Statistics. Springer.
Kallepalli, A., Kumar, A., Khoshelham, K., Nov. 2014. Entropy based determination of optimal principal components of Airborne Prism Experiment (APEX) imaging spectrometer data for improved land cover classification. ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 781–786.
Kang, X., Li, C., Li, S., Lin, H., 2018. Classification of hyperspectral images by gabor filtering based deep network. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 11 (4), 1166–1178.
Kang, X., Zhang, X., Li, S., Li, K., Li, J., Benediktsson, J.A., 2017. Hyperspectral anomaly detection with attribute and edge-preserving filters. IEEE Trans. Geosci. Remote Sens. 55 (10), 5600–5611.
Kang, X., Zhuo, B., Duan, P., 2018. Dual-path network-based hyperspectral image classification. IEEE Geosci. Remote Sens. Lett. 1–5.
Kang, X., Zhuo, B., Duan, P., 2019. Semi-supervised deep learning for hyperspectral image classification. Remote Sens. Lett. 10 (4), 353–362.
Karhunen, J., Raiko, T., Cho, K., 2015. Unsupervised Deep Learning: A Short Review.
Kaufmann, H., Segl, K., Guanter, L., Hofer, S., Foerster, K.-P., Stuffler, T., Mueller, A., Richter, R., Bach, H., Hostert, P., et al., 2008. Environmental mapping and analysis program (enmap)-recent advances and status. In: Geoscience and Remote Sensing Symposium, 2008. IGARSS 2008. IEEE International, vol. 4. IEEE, pp. IV–109.
Keshava, N., 2004. Distance metrics and band selection in hyperspectral processing with applications to material identification and spectral libraries. IEEE Tran. Geosci. Remote Sens. 42 (7), 1552–1565.
Kessy, A., Lewin, A., Strimmer, K., 2018. Optimal whitening and decorrelation. Am. Stat. 72 (4), 309–314.
Ketkar, N., 2017. Introduction to keras. In: Deep Learning with Python. Springer, pp. 97–111.
Khan, M.J., Khan, H.S., Yousaf, A., Khurshid, K., Abbas, A., 2018. Modern trends in hyperspectral image analysis: A review. IEEE Access 6, 14118–14129.
Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
Klambauer, G., Unterthiner, T., Mayr, A., Hochreiter, S., 2017. Self-normalizing neural networks. In: Advances in Neural Information Processing Systems. pp. 971–980.
Koch, G., Zemel, R., Salakhutdinov, R., 2015. Siamese neural networks for one-shot image recognition. In: ICML Deep Learning Workshop. Vol. 2. p.
Kokaly, R.F., Hoefen, T.M., Graham, G.E., Kelley, K.D., Johnson, M.R., Hubbard, B.E., Goldfarb, R.J., Buchhorn, M., Prakash, A., 2016. Mineral information at micron to kilometer scales: Laboratory, field, and remote sensing imaging spectrometer data from the orange hill porphyry copper deposit, alaska, usa. In: 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 5418–5421.
Kokaly, R.F., King, T.V., Hoefen, T.M., 2013. Surface mineral maps of afghanistan derived from hymap imaging spectrometer data, version 2. Tech. Rep., U.S. Geological Survey Data Series 787.
Koponen, S., Pulliainen, J., Kallio, K., Hallikainen, M., 2002. Lake water quality classification with airborne hyperspectral spectrometer and simulated meris data. Remote Sens. Environ. 79 (1), 51–59.
Kotsiantis, S.B., Zaharakis, I., Pintelas, P., 2007. Supervised machine learning: A review of classification techniques. Emerg. Artif. Intell. Appl. Comput. Eng. 160, 3–24.
Kotsiantis, S.B., Zaharakis, I.D., Pintelas, P.E., 2006. Machine learning: a review of classification and combining techniques. Artif. Intell. Rev. 26 (3), 159–190.
Koturwar, S., Merchant, S., 2017. Weight initialization of deep neural networks (dnns) using data statistics. arXiv preprint arXiv:1710.10570.
Krizhevsky, A., 5 2012. Learning multiple layers of features from tiny images. Tech. Rep., University of Toronto.
Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems. pp. 1097–1105.
Kruse, F., Boardman, J., Lefkoff, A., Young, J., Kierein-Young, K., Cocks, T., Jensen, R., Cocks, P., 2000. Hymap: an australian hyperspectral sensor solving global problems-results from usa hymap data acquisitions. In: Proc. of the 10th Australasian Remote Sensing and Photogrammetry Conference. pp. 18–23.
Kuching, S., 2007. The performance of maximum likelihood, spectral angle mapper, neural network and decision tree classifiers in hyperspectral image analysis. J. Comput. Sci. 3 (6), 419–423.
Kunkel, B., Blechinger, F., Lutz, R., Doerffer, R., van der Piepen, H., Schroder, M., 1988. ROSIS (Reflective Optics System Imaging Spectrometer) - A candidate instrument for polar platform missions. In: Seeley, J., Bowyer, S. (Eds.), Proc. SPIE 0868 Optoelectronic technologies for remote sensing from space, pp. 8.
Landgrebe, D., 2002. Hyperspectral image data analysis. IEEE Signal Process. Mag. 19 (1), 17–28.
Landgrebe, D.A., 2005. Signal Theory Methods in Multispectral Remote Sensing. Wiley-Blackwell.
Lange, J., Cavallaro, G., Götz, M., Erlingsson, E., Riedel, M., 2018. The influence of sampling methods on pixel-wise hyperspectral image classification with 3d convolutional neural networks. In: IGARSS 2018–2018 IEEE International Geoscience and Remote Sensing Symposium, pp. 2087–2090.
Larochelle, H., Bengio, Y., 2008. Classification using Discriminative Restricted Boltzmann Machines. In: Proceedings of the 25th international conference on Machine learning - ICML '08. p. 536.
Le, J.H., Yazdanpanah, A.P., Regentova, E.E., Muthukumar, V., 2015. A deep belief network for classifying remotely-sensed hyperspectral data. In: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Pavlidis, I., Feris, R., McGraw, T., Elendt, M., Kopper, R., Ragan, E., Ye, Z., Weber, G. (Eds.), Advances in Visual Computing. Springer International Publishing, Cham, pp. 682–692.
LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep Learning. Nature 521, 436–444.
Lecun, Y., Bottou, L., Bengio, Y., Haffner, P., 1998. Gradient-based learning applied to document recognition. Proc. IEEE 86 (11), 2278–2324.
Lee, C.A., Gasster, S.D., Plaza, A., Chang, C.-I., Huang, B., 2011. Recent developments in

high performance computing for remote sensing: A review. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 4 (3), 508–527.
Lee, H., Kwon, H., 2016. Contextual deep cnn based hyperspectral classification. In: 2016 IEEE International Geoscience and Remote Sensing Symposium IGARSS, pp. 3322–3325.
Lee, H., Kwon, H., 2017. Going deeper with contextual cnn for hyperspectral image classification. IEEE Trans. Image Process. 26 (10), 4843–4855.
Lei, D., Chen, X., Zhao, J., 2018. Opening the black box of deep learning. arXiv preprint arXiv:1805.08355.
Leng, J., Li, T., Bai, G., Dong, Q., Dong, H., 2016. Cube-cnn-svm: A novel hyperspectral image classification method. In: 2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI), pp. 1027–1034.
Li, C., Chen, C., Carlson, D., Carin, L., 2016. Preconditioned stochastic gradient langevin dynamics for deep neural networks. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence. AAAI'16. AAAI Press, pp. 1788–1794.
Li, J., June 2015. Active learning for hyperspectral image classification with a stacked autoencoders based neural network. In: 2015 7th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS). pp. 1–4.
Li, J., Bruzzone, L., Liu, S., 2015a. Deep feature representation for hyperspectral image classification. In: 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS). pp. 4951–4954.
Li, J., Zhao, X., Li, Y., Du, Q., Xi, B., Hu, J., 2018a. Classification of hyperspectral imagery using a new fully convolutional neural network. IEEE Geosci. Remote Sens. Lett. 15 (2), 292–296.
Li, S., Song, W., Fang, L., Chen, Y., Ghamisi, P., Benediktsson, J.A., 2019a. Deep learning for hyperspectral image classification: An overview. IEEE Trans. Geosci. Remote Sens.
Li, T., Leng, J., Kong, L., Guo, S., Bai, G., Wang, K., 2018b. Dcnr: deep cube cnn with random forest for hyperspectral image classification. Multimedia Tools Appl.
Li, T., Zhang, J., Zhang, Y., 2014. Classification of hyperspectral image based on deep belief networks. In: Proc. IEEE Int. Conf. Image Proces. pp. 5132–5136.
Li, W., Chen, C., Su, H., Du, Q., 2015b. Local binary patterns and extreme learning machine for hyperspectral imagery classification. IEEE Trans. Geosci. Remote Sens. 53 (7), 3681–3693.
Li, W., Du, Q., 2016. A survey on representation-based classification and detection in hyperspectral remote sensing imagery. Pattern Recogn. Lett. 83, 115–123.
Li, W., Wu, G., Zhang, F., Du, Q., 2017a. Hyperspectral image classification using deep pixel-pair features. IEEE Trans. Geosci. Remote Sens. 55 (2), 844–853.
Li, X., 2018. Preconditioned stochastic gradient descent. IEEE Trans. Neural Networks Learn. Syst. 29 (5), 1454–1466.
Li, Y., Xie, W., Li, H., 2017b. Hyperspectral image reconstruction by deep convolutional neural network for classification. Pattern Recogn. 63, 371–383.
Li, Y., Zhang, H., Shen, Q., 2017c. Spectral-spatial classification of hyperspectral imagery with 3d convolutional neural network. Remote Sens. 9 (1), 67.
Li, Z., Huang, L., He, J., 2019b. A multiscale deep middle-level feature fusion network for hyperspectral classification. Remote Sens. 11 (6), 695.
Liang, H., Li, Q., 2016. Hyperspectral imagery classification using sparse representations of convolutional neural network features. Remote Sens. 8 (2), 99.
Liang, J., Zhou, J., Qian, Y., Wen, L., Bai, X., Gao, Y., 2017. On the sampling strategy for evaluation of spectral-spatial methods in hyperspectral image classification. IEEE Trans. Geosci. Remote Sens. 55 (2), 862–880.
Liang, M., Jiao, L., Yang, S., Liu, F., Hou, B., Chen, H., 2018. Deep multiscale spectral-spatial feature fusion for hyperspectral images classification. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 11 (8), 2911–2924.
Lin, M., Chen, Q., Yan, S., 2013a. Network in network. arXiv preprint arXiv:1312.4400.
Lin, Z., Chen, Y., Zhao, X., Wang, G., Dec 2013b. Spectral-spatial classification of hyperspectral image using autoencoders. In: 2013 9th International Conference on Information, Communications Signal Processing. pp. 1–5.
Lipton, Z.C., 2016. The mythos of model interpretability. arXiv preprint arXiv:1606.03490.
Liu, B., Yu, X., Zhang, P., Tan, X., Yu, A., Xue, Z., 2017a. A semi-supervised convolutional neural network for hyperspectral image classification. Remote Sens. Lett. 8 (9), 839–848.
Liu, B., Yu, X., Zhang, P., Yu, A., Fu, Q., Wei, X., 2018. Supervised deep feature extraction for hyperspectral image classification. IEEE Trans. Geosci. Remote Sens. 56 (4), 1909–1921.
Liu, M., Shi, J., Li, Z., Li, C., Zhu, J., Liu, S., 2017b. Towards better analysis of deep convolutional neural networks. IEEE Trans. Visual. Comput. Graphics 23 (1), 91–100.
Liu, P., Zhang, H., Eom, K.B., 2017c. Active deep learning for classification of hyperspectral images. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 10 (2), 712–724.
Liu, Q., Zhou, F., Hang, R., Yuan, X., 2017d. Bidirectional-convolutional lstm based spectral-spatial feature learning for hyperspectral image classification. Remote Sens. 9 (12), 1330.
Liu, X., Van De Weijer, J., Bagdanov, A.D., 2019. Exploiting unlabeled data in cnns by self-supervised learning to rank. IEEE Trans. Pattern Anal. Machine Intell.
Liu, Y., Cao, G., Sun, Q., Siegel, M., 2015. Hyperspectral classification via deep networks and superpixel segmentation. Int. J. Remote Sens. 36 (13), 3459–3482.
Long, J., Shelhamer, E., Darrell, T., June 2015a. Fully convolutional networks for semantic segmentation. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3431–3440.
Long, M., Cao, Y., Wang, J., Jordan, M.I., 2015b. Learning transferable features with deep adaptation networks. arXiv preprint arXiv:1502.02791.
Lu, D., Weng, Q., 2007. A survey of image classification methods and techniques for improving classification performance. Int. J. Remote Sens. 28 (5), 823–870.
Lucas, R., Rowlands, A., Niemann, O., Merton, R., 2004. Advanced Image Processing Techniques for Remotely Sensed Hyperspectral Data. Springer Berlin Heidelberg, Berlin, Heidelberg.
Lulla, V., 2009. Hyperspectral applications in urban geography. In: Gatrell, J.D., Jensen, R.R. (Eds.), Planning and Socioeconomic Applications. Springer, Netherlands, Dordrecht, pp. 79–86.
Luo, H., 2018. Shorten spatial-spectral rnn with parallel-gru for hyperspectral image classification. arXiv preprint arXiv:1810.12563.
Luo, Y., Zou, J., Yao, C., Zhao, X., Li, T., Bai, G., 2018. Hsi-cnn: A novel convolution neural network for hyperspectral image. In: 2018 International Conference on Audio, Language and Image Processing (ICALIP), pp. 464–469.
Lyu, H., Lu, H., Mou, L., 2016. Learning a transferable change rule from a recurrent neural network for land cover change detection. Remote Sens. 8 (6), 506.
Ma, N., Peng, Y., Wang, S., Leong, P., 2018a. An unsupervised deep hyperspectral anomaly detector. Sensors 18 (3), 693.
Ma, W., Yang, Q., Wu, Y., Zhao, W., Zhang, X., 2019. Double-branch multi-attention mechanism network for hyperspectral image classification. Remote Sens. 11 (11), 1307.
Ma, X., Fu, A., Wang, J., Wang, H., Yin, B., 2018b. Hyperspectral image classification based on deep deconvolution network with skip architecture. IEEE Trans. Geosci. Remote Sens. 56 (8), 4781–4791.
Ma, X., Wang, H., Geng, J., 2016a. Spectral-spatial classification of hyperspectral image based on deep auto-encoder. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 9 (9), 4073–4085.
Ma, X., Wang, H., Geng, J., Wang, J., July 2016b. Hyperspectral image classification with small training set by deep network and relative distance prior. In: 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS). pp. 3282–3285.
Ma, X., Wang, H., Wang, J., 2016c. Semisupervised classification for hyperspectral image based on multi-decision labeling and deep feature learning. ISPRS J. Photogramm. Remote Sens. 120, 99–107.
Maas, A.L., Hannun, A.Y., Ng, A.Y., 2013. Rectifier nonlinearities improve neural network acoustic models. In: Proc. icml. vol. 30. p. 3.
Maaten, L.v.d., Hinton, G., 2008. Visualizing data using t-sne. J. Machine Learn. Res. 9 (Nov), 2579–2605.
MacKay, D.J.C., 1992. Information-based objective functions for active data selection. Neural Comput. 4 (4), 590–604.
Mahendran, A., Vedaldi, A., 2015. Understanding deep image representations by inverting them. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5188–5196.
Mahesh, S., Jayas, D., Paliwal, J., White, N., 2015. Hyperspectral imaging to classify and monitor quality of agricultural materials. J. Stored Prod. Res. 61, 17–26.
Maji, P., Mullins, R., 2018. On the reduction of computational complexity of deep convolutional neural networks. Entropy 20 (4), 305.
Makantasis, K., Karantzalos, K., Doulamis, A., Doulamis, N., 2015. Deep supervised learning for hyperspectral data classification through convolutional neural networks. In: 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 4959–4962.
Man, Q., Dong, P., Guo, H., 2015. Pixel- and feature-level fusion of hyperspectral and lidar data for urban land-use classification. Int. J. Remote Sens. 36 (6), 1618–1644.
Martens, J., Sutskever, I., 2012. Training deep and recurrent networks with hessian-free optimization. In: Montavon, G., Orr, G.B., Müller, K.-R. (Eds.), Neural Networks: Tricks of the Trade, Second Edition. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 479–535.
Mazhari, N., Malekzadeh Shafaroudi, A., Ghaderi, M., 2017. Detecting and mapping different types of iron mineralization in sangan mining region, ne iran, using satellite image and airborne geophysical data. Geosci. J. 21 (1), 137–148.
McInnes, L., Healy, J., Melville, J., 2018. Umap: Uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:1802.03426.
Mei, S., Ji, J., Bi, Q., Hou, J., Du, Q., Li, W., 2016. Integrating spectral and spatial information into deep convolutional neural networks for hyperspectral classification. In: 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 5067–5070.
Mei, S., Ji, J., Geng, Y., Zhang, Z., Li, X., Du, Q., 2019a. Unsupervised spatial-spectral feature learning by 3d convolutional autoencoder for hyperspectral classification. IEEE Trans. Geosci. Remote Sens.
Mei, S., Ji, J., Hou, J., Li, X., Du, Q., 2017. Learning sensor-specific spatial-spectral features of hyperspectral images via convolutional neural networks. IEEE Trans. Geosci. Remote Sens. 55 (8), 4520–4533.
Mei, X., Pan, E., Ma, Y., Dai, X., Huang, J., Fan, F., Du, Q., Zheng, H., Ma, J., 2019b. Spectral-spatial attention networks for hyperspectral image classification. Remote Sens. 11 (8), 963.
Melgani, F., Bruzzone, L., 2004. Classification of hyperspectral remote sensing images with support vector machines. IEEE Trans. Geosci. Remote Sens. 42 (8), 1778–1790.
Min, E., Guo, X., Liu, Q., Zhang, G., Cui, J., Long, J., 2018. A survey of clustering with deep learning: From the perspective of network architecture. IEEE Access 6, 39501–39514.
Molchanov, D., Ashukha, A., Vetrov, D., 2017. Variational dropout sparsifies deep neural networks. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, pp. 2498–2507.
Mookambiga, A., Gomathi, V., 2016. Comprehensive review on fusion techniques for spatial information enhancement in hyperspectral imagery. Multidimension. Syst. Signal Process. 27 (4), 863–889.
Mou, L., Bruzzone, L., Zhu, X.X., 2019. Learning spectral-spatial-temporal features via a recurrent convolutional neural network for change detection in multispectral imagery. IEEE Trans. Geosci. Remote Sens. 57 (2), 924–935.
Mou, L., Ghamisi, P., Zhu, X.X., 2017. Deep recurrent neural networks for hyperspectral image classification. IEEE Trans. Geosci. Remote Sens. 55 (7), 3639–3655.
Mou, L., Ghamisi, P., Zhu, X.X., 2018. Unsupervised spectral-spatial feature learning via deep residual conv-deconv network for hyperspectral image classification. IEEE

Trans. Geosci. Remote Sens. 56 (1), 391–406.
Mouroulis, P., Van Gorp, B., Green, R.O., Dierssen, H., Wilson, D.W., Eastwood, M., Boardman, J., Gao, B.-C., Cohen, D., Franklin, B., et al., 2014. Portable remote imaging spectrometer coastal ocean sensor: design, characteristics, and first flight results. Appl. Opt. 53 (7), 1363–1380.
Mughees, A., Tao, L., 2016. Efficient deep auto-encoder learning for the classification of hyperspectral images. In: 2016 International Conference on Virtual Reality and Visualization (ICVRV), pp. 44–51.
Mura, M.D., Benediktsson, J.A., Waske, B., Bruzzone, L., 2010. Morphological attribute profiles for the analysis of very high resolution images. IEEE Trans. Geosci. Remote Sens. 48 (10), 3747–3762.
Murugan, P., 2017. Feed forward and backward run in deep convolution neural network. arXiv preprint arXiv:1711.03278.
Murugan, P., Durairaj, S., 2017. Regularization and optimization strategies in deep convolutional neural network. arXiv preprint arXiv:1712.04711.
Nair, V., Hinton, G.E., 2010. Rectified Linear Units Improve Restricted Boltzmann Machines. In: Johannes Fürnkranz and Thorsten Joachims (Ed.), Proceedings of the 27th International Conference on Machine Learning (ICML-10). Omnipress, pp. 807–814.
Nam, H., Kim, H.-E., 2018. Batch-instance normalization for adaptively style-invariant neural networks. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 31. Curran Associates, Inc., pp. 2558–2567.
Narumalani, S., Mishra, D.R., Wilson, R., Reece, P., Kohler, A., 2009. Detecting and mapping four invasive species along the floodplain of north Platte river, Nebraska. Weed Technol. 23 (1), 99–107.
Newell, A., Yang, K., Deng, J., 2016. Stacked hourglass networks for human pose estimation. In: European Conference on Computer Vision. Springer, pp. 483–499.
Nguyen, A., Yosinski, J., Clune, J., 2015. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 427–436.
Nguyen, Q., Hein, M., 2018. Optimization landscape and expressivity of deep cnns. In: International Conference on Machine Learning, pp. 3727–3736.
Nogueira, K., Penatti, O.A., dos Santos, J.A., 2017. Towards better exploiting convolutional neural networks for remote sensing scene classification. Pattern Recogn. 61, 539–556.
Okan, A., Özdemir, B., Gedik, B.E., Yasemin, C., Çetin, Y., 2014. Hyperspectral classification using stacked autoencoders with deep learning. In: 2014 6th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS). pp. 1–4.
Olmanson, L.G., Brezonik, P.L., Bauer, M.E., 2013. Airborne hyperspectral remote sensing to assess spatial distribution of water quality characteristics in large rivers: The mississippi river and its tributaries in minnesota. Remote Sens. Environ. 130, 254–265.
Pan, S.J., Yang, Q., 2010. A survey on transfer learning. IEEE Trans. Knowledge Data Eng. 22 (10), 1345–1359.
Pan, X., Luo, P., Shi, J., Tang, X., 2018. Two at once: Enhancing learning and generalization capacities via ibn-net. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 464–479.
Paoletti, M., Haut, J., Plaza, J., Plaza, A., 2018a. Deep&dense convolutional neural network for hyperspectral image classification. Remote Sens. 10 (9), 1454.
Paoletti, M.E., Haut, J.M., Fernandez-Beltran, R., Plaza, J., Plaza, A.J., Pla, F., 2018b. Capsule networks for hyperspectral image classification. IEEE Trans. Geosci. Remote Sens. 1–16.
Paoletti, M.E., Haut, J.M., Fernandez-Beltran, R., Plaza, J., Plaza, A.J., Pla, F., 2018c. Deep pyramidal residual networks for spectral-spatial hyperspectral image classification. IEEE Trans. Geosci. Remote Sens. 1–15.
Paoletti, M.E., Haut, J.M., Plaza, J., Plaza, A., 2017a. A new deep convolutional neural network for fast hyperspectral image classification. ISPRS J. Photogramm. Remote Sens.
Paoletti, M.E., Haut, J.M., Plaza, J., Plaza, A., Liu, Q., Hang, R., July 2017b. Multicore implementation of the multi-scale adaptive deep pyramid matching model for remotely sensed image classification. In: 2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS). pp. 2247–2250.
Paoletti, M.E., m. Haut, J., Plaza, J., Plaza, A., July 2018. An investigation on self-normalized deep neural networks for hyperspectral image classification. In: IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium. pp. 3607–3610.
Park, S., Kwak, N., 2017. Analysis on the dropout effect in convolutional neural networks. In: Lai, S.-H., Lepetit, V., Nishino, K., Sato, Y. (Eds.), Computer Vision – ACCV 2016. Springer International Publishing, Cham, pp. 189–204.
Patricia, N., Caputo, B., 2014. Learning to learn, from transfer learning to domain adaptation: A unifying perspective. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1442–1449.
Paul, S., Kumar, D.N., 2018. Spectral-spatial classification of hyperspectral data with mutual information based segmented stacked autoencoder approach. ISPRS J. Photogramm. Remote Sens. 138, 265–280.
Pearlman, J.S., Barry, P.S., Segal, C.C., Shepanski, J., Beiso, D., Carman, S.L., 2003. Hyperion, a space-based imaging spectrometer. IEEE Trans. Geosci. Remote Sens. 41 (6), 1160–1173.
Pedamonti, D., 2018. Comparison of non-linear activation functions for deep neural networks on mnist classification task. arXiv preprint arXiv:1804.02763.
Peerbhay, K.Y., Mutanga, O., Ismail, R., 2015. Random forests unsupervised classification: The detection and mapping of solanum mauritianum infestations in plantation forestry using hyperspectral data. IEEE J. Sel. Top. Appl. Earth Obser. Remote Sens. 8 (6), 3107–3122.
Penttilä, J., 2017. A method for anomaly detection in hyperspectral images, using deep convolutional autoencoders. Master's thesis. University of Jyväskylä.
Petersson, H., Gustafsson, D., Bergstrom, D., 2016. Hyperspectral image analysis using deep learning—a review. In: 2016 Sixth International Conference on Image Processing Theory, Tools and Applications (IPTA). IEEE, pp. 1–6.
Pezeshki, M., Fan, L., Brakel, P., Courville, A., Bengio, Y., 2016. Deconstructing the ladder network architecture. In: International Conference on Machine Learning, pp. 2368–2376.
Pignatti, S., Palombo, A., Pascucci, S., Romano, F., Santini, F., Simoniello, T., Umberto, A., Vincenzo, C., Acito, N., Diani, M., et al., 2013. The prisma hyperspectral mission: Science activities and opportunities for agriculture and land monitoring. In: 2013 IEEE International Geoscience and Remote Sensing Symposium-IGARSS. IEEE, pp. 4558–4561.
Plaut, E., 2018. From principal subspaces to principal components with linear autoencoders. CoRR abs/1804.10253. URL http://arxiv.org/abs/1804.10253.
Plaza, A., Benediktsson, J.A., Boardman, J.W., Brazile, J., Bruzzone, L., Camps-Valls, G., Chanussot, J., Fauvel, M., Gamba, P., Gualtieri, A., Marconcini, M., Tilton, J.C., Trianni, G., 2009. Recent advances in techniques for hyperspectral image processing. Remote Sens. Environ. 113 (1), S110–S122.
Plaza, A., Chang, C.-I., 2008. Clusters versus fpga for parallel processing of hyperspectral imagery. Int. J. High Performance Comput. Appl. 22 (4), 366–385.
Plaza, A., Du, Q., Chang, Y.-L., King, R.L., 2011a. High performance computing for hyperspectral remote sensing. IEEE J. Sel. Top. Appl. Earth Obser. Remote Sens. 4 (3), 528–544.
Plaza, A., Plaza, J., Paz, A., Sanchez, S., 2011b. Parallel hyperspectral image and signal processing [applications corner]. IEEE Signal Process. Mag. 28 (3), 119–126.
Qiu, F., Jensen, J., 2004. Opening the black box of neural networks for remote sensing image classification. Int. J. Remote Sens. 25 (9), 1749–1768.
Qiu, Q., Wu, X., Liu, Z., Tang, B., Zhao, Y., Wu, X., Zhu, H., Xin, Y., 2017. Survey of supervised classification techniques for hyperspectral images. Sensor Rev. 37 (3), 371–382.
Quirita, V.A.A., da Costa, G.A.O.P., Happ, P.N., Feitosa, R.Q., Ferreira, R.d.S., Oliveira, D.A.B., Plaza, A., 2017. A new cloud computing architecture for the classification of remote sensing data. IEEE J. Sel. Top. Appl. Earth Obser. Remote Sens. 10 (2), 409–416.
Ramachandran, P., Zoph, B., Le, Q.V., 2017. Swish: a self-gated activation function. arXiv preprint arXiv:1710.05941 7.
Ran, L., Zhang, Y., Wei, W., Zhang, Q., 2017. A hyperspectral image classification framework with spatial pixel pair features. Sensors 17 (10), 2421.
Randhe, P.H., Durbha, S.S., Younan, N.H., Aug 2016. Embedded high performance computing for on-board hyperspectral image classification. In: 2016 8th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS). pp. 1–5.
Rasmus, A., Berglund, M., Honkala, M., Valpola, H., Raiko, T., 2015. Semi-supervised learning with ladder networks. In: Advances in neural information processing systems. pp. 3546–3554.
Rasti, B., Scheunders, P., Ghamisi, P., Licciardi, G., Chanussot, J., 2018. Noise reduction in hyperspectral imagery: Overview and application. Remote Sens. 10 (3), 482.
Ratle, F., Camps-Valls, G., Weston, J., 2010. Semisupervised neural networks for efficient hyperspectral image classification. IEEE Trans. Geosci. Remote Sens. 48 (5), 2271–2282.
Rauber, P.E., Fadel, S.G., Falcao, A.X., Telea, A.C., 2017. Visualizing the hidden activity of artificial neural networks. IEEE Trans. Visualizat. Comput. Graphics 23 (1), 101–110.
Ravanelli, M., Bengio, Y., 2018. Interpretable convolutional filters with sincnet. arXiv preprint arXiv:1811.09725.
Resmini, R.G., Kappus, M.E., Aldrich, W.S., Harsanyi, J.C., Anderson, M., 1997. Mineral mapping with hyperspectral digital imagery collection experiment (hydice) sensor data at cuprite, nevada, u.s.a. Int. J. Remote Sens. 18 (7), 1553–1570.
Richter, R., 2005. Hyperspectral sensors for military applications. Tech. Rep., German Aerospace Center Wessling (DLR), Wessling (Germany).
Rickard, L.J., Basedow, R.W., Zalewski, E.F., Silverglate, P.R., Landers, M., 1993. Hydice: An airborne system for hyperspectral imaging. In: Imaging Spectrometry of the Terrestrial Environment. vol. 1937. International Society for Optics and Photonics, pp. 173–180.
Rifai, S., Vincent, P., Muller, X., Glorot, X., Bengio, Y., 2011. Contractive auto-encoders: Explicit invariance during feature extraction. In: Proceedings of the 28th International Conference on International Conference on Machine Learning. Omnipress, pp. 833–840.
Roberts, D.A., Dennison, P.E., Gardner, M.E., Hetzel, Y., Ustin, S.L., Lee, C.T., 2003. Evaluation of the potential of hyperion for fire danger assessment by comparison to the airborne visible/infrared imaging spectrometer. IEEE Trans. Geosci. Remote Sens. 41 (6), 1297–1310.
Roberts, D.A., Quattrochi, D.A., Hulley, G.C., Hook, S.J., Green, R.O., 2012. Synergies between vswir and tir data for the urban environment: An evaluation of the potential for the hyperspectral infrared imager (hyspiri) decadal survey mission. Remote Sens. Environ. 117, 83–101.
Rodríguez-Pérez, J.R., Riaño, D., Carlisle, E., Ustin, S., Smart, D.R., 2007. Evaluation of hyperspectral reflectance indexes to detect grapevine water status in vineyards. Am. J. Enology Viticulture 58 (3), 302–317.
Romero, A., Gatta, C., Camps-Valls, G., 2016. Unsupervised deep feature extraction for remote sensing image classification. IEEE Trans. Geosci. Remote Sens. 54 (3), 1349–1362.
Romero, A., Radeva, P., Gatta, C., 2015. Meta-parameter free unsupervised sparse feature learning. IEEE Trans. Pattern Anal. Mach. Intell. 37 (8), 1716–1722.
Roodposhti, M.S., Aryal, J., Lucieer, A., Bryan, B.A., 2019. Uncertainty assessment of

hyperspectral image classification: Deep learning vs. random forest. Entropy 21 (1), 78.
Roy, S.K., Krishna, G., Dubey, S.R., Chaudhuri, B.B., 2019. Hybridsn: Exploring 3d–2d cnn feature hierarchy for hyperspectral image classification. arXiv preprint arXiv:1902.06701.
Rußwurm, M., Körner, M., 2017. Temporal vegetation modelling using long short-term memory networks for crop identification from medium-resolution multi-spectral satellite images. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 1496–1504.
Sabale, S.P., Jadhav, C.R., 2015. Hyperspectral image classification methods in remote sensing - a review. In: 2015 International Conference on Computing Communication Control and Automation, pp. 679–683.
Sabalel, S., Jadhav, C., 2014. Supervised, unsupervised, and semisupervised classification methods for hyperspectral image classification-a review. Int. J. Sci. Res. (IJSR) 3 (12).
Sabour, S., Frosst, N., Hinton, G.E., 2017. Dynamic routing between capsules. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 30. Curran Associates, Inc., pp. 3856–3866.
Salimans, T., Kingma, D.P., 2016. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In: Advances in Neural Information Processing Systems. pp. 901–909.
Salman, M., Yüksel, S.E., May 2016. Hyperspectral data classification using deep convolutional neural networks. In: 2016 24th Signal Processing and Communication Application Conference (SIU). pp. 2129–2132.
Sánchez, S., Ramalho, R., Sousa, L., Plaza, A., 2015. Real-time implementation of remotely sensed hyperspectral image unmixing on gpus. J. Real-Time Image Proc. 10 (3), 469–483.
Santurkar, S., Tsipras, D., Ilyas, A., Madry, A., 2018. How does batch normalization help optimization? In: Advances in Neural Information Processing Systems. pp. 2483–2493.
Sanz, C., 2001. Der (dynamic evidential reasoning), applied to the classification of hyperspectral images. In: IGARSS 2001. Scanning the Present and Resolving the Future. Proceedings. IEEE 2001 International Geoscience and Remote Sensing Symposium (Cat. No. 01CH37217), vol. 4. IEEE, pp. 1904–1906.
Savage, S.H., Levy, T.E., Jones, I.W., 2012. Prospects and problems in the use of hyperspectral imagery for archaeological remote sensing: a case study from the faynan copper mining district, jordan. J. Archaeol. Sci. 39 (2), 407–420.
Scafutto, R.D.M., de Souza Filho, C.R., de Oliveira, W.J., 2017. Hyperspectral remote sensing detection of petroleum hydrocarbons in mixtures with mineral substrates: Implications for onshore exploration and monitoring. ISPRS J. Photogramm. Remote Sens. 128, 146–157.
Scherer, D., Müller, A., Behnke, S., 2010. Evaluation of pooling operations in convolutional architectures for object recognition. In: Diamantaras, K., Duch, W., Iliadis, L.S. (Eds.), Artificial Neural Networks – ICANN 2010. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 92–101.
Schmidhuber, J., 2015. Deep learning in neural networks: An overview. Neural Networks 61, 85–117.
Sellami, A., Farah, M., Farah, I.R., Solaiman, B., 2019. Hyperspectral imagery classification based on semi-supervised 3-d deep neural network and adaptive band selection. Expert Syst. Appl. 129, 246–259.
Seydgar, M., Alizadeh Naeini, A., Zhang, M., Li, W., Satari, M., 2019. 3-d convolution-recurrent networks for spectral-spatial classification of hyperspectral images. Remote Sens. 11 (7), 883.
Shaham, U., Stanton, K., Li, H., Nadler, B., Basri, R., Kluger, Y., 2018. Spectralnet: Spectral clustering using deep neural networks. arXiv preprint arXiv:1801.01587.
Shamsolmoali, P., Zareapoor, M., Yang, J., 2018. Convolutional neural network in network (cnnin): hyperspectral image classification and dimensionality reduction. IET Image Proc.
Shang, X., Chisholm, L.A., 2014. Classification of australian native forest species using hyperspectral remote sensing and machine-learning classification algorithms. IEEE J. Sel. Top. Appl. Earth Obser. Remote Sens. 7 (6), 2481–2489.
Sharma, A., Liu, X., Yang, X., 2018. Land cover classification from multi-temporal, multi-spectral remotely sensed imagery using patch-based recurrent neural networks. Neural Networks 105, 346–355.
Shi, C., Pun, C.-M., 2018. Multi-scale hierarchical recurrent neural networks for hyperspectral image classification. Neurocomputing 294, 82–93.
Shi, C., Wang, L., 2014. Incorporating spatial information in spectral unmixing: A review. Remote Sens. Environ. 149, 70–87.
Shi, X.-Z., Aspandiar, M., Oldmeadow, D., 2014. Using hyperspectral data and plsr modelling to assess acid sulphate soil in subsurface. J. Soils Sediments 14 (5), 904–916.
Shwartz-Ziv, R., Tishby, N., 2017. Opening the black box of deep neural networks via information. arXiv preprint arXiv:1703.00810.
Sidorov, O., Hardeberg, J.Y., 2019. Deep hyperspectral prior: Denoising, inpainting, super-resolution. arXiv preprint arXiv:1902.00301.
Signoroni, A., Savardi, M., Baronio, A., Benini, S., 2019. Deep learning meets hyperspectral image analysis: A multidisciplinary review. J. Imag. 5 (5), 52.
Sima, C., Dougherty, E.R., 2008. The peaking phenomenon in the presence of feature-selection. Pattern Recogn. Lett. 29 (11), 1667–1674.
Simonyan, K., Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.
Simpson, A.J., 2015. Dither is better than dropout for regularising deep neural networks. arXiv preprint arXiv:1508.04826.
Slavkovikj, V., Verstockt, S., De Neve, W., Van Hoecke, S., Van de Walle, R., 2015. Hyperspectral image classification with convolutional neural networks. In: Proceedings of the 23rd ACM International Conference on Multimedia. ACM, pp. 1159–1162.
Smolensky, P., 1986. Information processing in dynamical systems: foundations of harmony theory. In: In: David, E., McLelland, J.L. (Eds.), Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol. 6. MIT Press, pp. 194–281 Ch. 6.
Song, B., Li, J., Mura, M.D., Li, P., Plaza, A., Bioucas-Dias, J.M., Benediktsson, J.A., Chanussot, J., 2014. Remotely sensed image classification using sparse representations of morphological attribute profiles. IEEE Trans. Geosci. Remote Sens. 52 (8), 5122–5136.
Song, W., Li, S., Fang, L., Lu, T., 2018. Hyperspectral image classification with deep feature fusion network. IEEE Trans. Geosci. Remote Sens. 56 (6), 3173–3184.
Sonoda, S., Murata, N., 2017. Neural network with unbounded activation functions is universal approximator. Appl. Comput. Harmonic Anal. 43 (2), 233–268.
Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M.A., 2014. Striving for simplicity: The all convolutional net. CoRR abs/1412.6806. URL http://arxiv.org/abs/1412.6806.
Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2014. Dropout: A simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. 15, 1929–1958.
Srivastava, R.K., Greff, K., Schmidhuber, J., 2015. Training very deep networks. In: Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 28. Curran Associates, Inc., pp. 2377–2385.
Stein, D.W.J., Beaven, S.G., Hoff, L.E., Winter, E.M., Schaum, A.P., Stocker, A.D., 2002. Anomaly detection from hyperspectral imagery. IEEE Signal Process. Mag. 19 (1), 58–69.
Strachan, I.B., Pattey, E., Boisvert, J.B., 2002. Impact of nitrogen and environmental conditions on corn as detected by hyperspectral reflectance. Remote Sens. Environ. 80 (2), 213–224.
Su, J., Vargas, D.V., Sakurai, K., 2019. One pixel attack for fooling deep neural networks. IEEE Trans. Evol. Comput.
Sun, L., Wu, Z., Liu, J., Xiao, L., Wei, Z., 2015. Supervised spectral–spatial hyperspectral image classification with weighted markov random fields. IEEE Trans. Geosci. Remote Sens. 53 (3), 1490–1503.
Sutskever, I., Martens, J., Dahl, G., Hinton, G., 17–19 Jun 2013. On the importance of initialization and momentum in deep learning. In: Dasgupta, S., McAllester, D. (Eds.), Proceedings of the 30th International Conference on Machine Learning. vol. 28 of Proceedings of Machine Learning Research. PMLR, Atlanta, Georgia, USA, pp. 1139–1147.
Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., 2015. Going deeper with convolutions. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1–9.
Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z., 2016. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826.
Tan, K., Wu, F., Du, Q., Du, P., Chen, Y., 2019. A parallel gaussian–bernoulli restricted boltzmann machine for mining area classification with hyperspectral imagery. IEEE J. Sel. Top. Appl. Earth Obser. Remote Sens. 12 (2), 627–636.
Tao, C., Pan, H., Li, Y., Zou, Z., 2015. Unsupervised spectral-spatial feature learning with stacked sparse autoencoder for hyperspectral imagery classification. IEEE Geosci. Remote Sens. Lett. 12 (12), 2438–2442.
Tarabalka, Y., Benediktsson, J.A., Chanussot, J., 2009. Spectral-spatial classification of hyperspectral imagery based on partitional clustering techniques. IEEE Trans. Geosci. Remote Sens. 47 (8), 2973–2987.
Tarabalka, Y., Fauvel, M., Chanussot, J., Benediktsson, J.A., 2010. Svm-and mrf-based method for accurate classification of hyperspectral images. IEEE Geosci. Remote Sens. Lett. 7 (4), 736–740.
Teke, M., Deveci, H.S., Haliloglu, O., Gürbüz, S.Z., Sakarya, U., 2013. A short survey of hyperspectral remote sensing applications in agriculture. In: 2013 6th International Conference on Recent Advances in Space Technologies (RAST), pp. 171–176.
Theodoridis, S., Koutroumbas, K., 2003. Pattern Recognition. Elsevier Science.
Tian, K., Zhou, S., Guan, J., 2017. Deepcluster: A general clustering framework based on deep learning. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer, pp. 809–825.
Tompson, J., Goroshin, R., Jain, A., LeCun, Y., Bregler, C., 2015. Efficient object localization using convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 648–656.
Transon, J., d'Andrimont, R., Maugnard, A., Defourny, P., 2017. Survey of current hyperspectral earth observation applications from space and synergies with sentinel-2. In: 2017 9th International Workshop on the Analysis of Multitemporal Remote Sensing Images (MultiTemp), pp. 1–8.
Transon, J., d'Andrimont, R., Maugnard, A., Defourny, P., 2018. Survey of hyperspectral earth observation applications from space in the sentinel-2 context. Remote Sens. 10 (2).
Tuia, D., Camps-Valls, G., Nov 2009. Recent advances in remote sensing image processing. In: 2009 16th IEEE International Conference on Image Processing (ICIP). pp. 3705–3708.
Tuia, D., Flamary, R., Courty, N., 2015. Multiclass feature learning for hyperspectral image classification: Sparse and hierarchical solutions. ISPRS J. Photogramm. Remote Sens. 105, 272–285.
Ulyanov, D., Lebedev, V., Vedaldi, A., Lempitsky, V., 2016a. Texture networks: Feed-forward synthesis of textures and stylized images. In: Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48. ICML'16. JMLR.org, pp. 1349–1357.
Ulyanov, D., Vedaldi, A., Lempitsky, V., 2016b. Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022.
Ustin, S.L., DiPietro, D., Olmstead, K., Underwood, E., Scheer, G.J., June 2002a. Hyperspectral remote sensing for invasive species detection and mapping. In: IEEE

International Geoscience and Remote Sensing Symposium. vol. 3. pp. 1658–1660 vol 3.
Ustin, S.L., Roberts, D.A., Gardner, M., Dennison, P., 2002b. Evaluation of the potential of hyperion data to estimate wildfire hazard in the santa ynez front range, santa barbara, california. In: IEEE International Geoscience and Remote Sensing Symposium. vol. 2. pp. 796–798 vol 2.
Vane, G., Evans, D.L., Kahle, A.B., 1989. Recent advances in airborne terrestrial remote sensing with the Nasa airborne visible/infrared imaging spectrometer (aviris), airborne synthetic aperture radar (sar), and thermal infrared multispectral scanner (tims). In: 12th Canadian Symposium on Remote Sensing Geoscience and Remote Sensing Symposium. pp. 942–943.
Varshney, P., Arora, M., 2004. Advanced Image Processing Techniques for Remotely Sensed Hyperspectral Data. Springer.
Venkatesan, R., Prabu, S., 2019. Hyperspectral image features classification using deep learning recurrent neural networks. J. Med. Syst. 43 (7), 216.
Veraverbeke, S., Dennison, P., Gitas, I., Hulley, G., Kalashnikova, O., Katagis, T., Kuai, L., Meng, R., Roberts, D., Stavros, N., 2018. Hyperspectral remote sensing of fire: State-of-the-art and future perspectives. Remote Sens. Environ. 216, 105–121.
Wan, L., Zeiler, M., Zhang, S., Cun, Y.L., Fergus, R., 17–19 Jun 2013. Regularization of neural networks using dropconnect. In: Dasgupta, S., McAllester, D. (Eds.), Proceedings of the 30th International Conference on Machine Learning. Vol. 28 of Proceedings of Machine Learning Research. PMLR, Atlanta, Georgia, USA, pp. 1058–1066.
Wan, X., Zhao, C., Wang, Y., Liu, W., 2017. Stacked sparse autoencoder in hyperspectral data classification using spectral-spatial, higher order statistics and multifractal spectrum features. Infrared Phys. Technol. 86, 77–89.
Wang, C., Liu, Y., Bai, X., Tang, W., Lei, P., Zhou, J., 2017a. Deep residual convolutional neural network for hyperspectral image super-resolution. In: Zhao, Y., Kong, X., Taubman, D. (Eds.), Image and Graphics. Springer International Publishing, Cham, pp. 370–380.
Wang, C., Zhang, P., Zhang, Y., Zhang, L., Wei, W., 2016. A multi-label hyperspectral image classification method with deep learning features. In: Proceedings of the International Conference on Internet Multimedia Computing and Service. ACM, pp. 127–131.
Wang, L., Zhang, J., Liu, P., Choo, K.-K.R., Huang, F., 2017b. Spectral–spatial multi-feature-based deep learning for hyperspectral remote sensing image classification. Soft. Comput. 21 (1), 213–221.
Wang, Q., Li, Q., Liu, H., Wang, Y., Zhu, J., Oct 2014. An improved isodata algorithm for hyperspectral image classification. In: 2014 7th International Congress on Image and Signal Processing. pp. 660–664.
Wang, W., Dou, S., Jiang, Z., Sun, L., 2018a. A fast dense spectral–spatial convolution network framework for hyperspectral images classification. Remote Sens. 10 (7), 1068.
Wang, W., Dou, S., Wang, S., 2019. Alternately updated spectral–spatial convolution network for the classification of hyperspectral images. Remote Sens. 11 (15), 1794.
Wang, Y., Jiang, Y., Wu, Y., Zhou, Z.-H., 2010. Multi-manifold clustering. In: Pacific Rim International Conference on Artificial Intelligence. Springer, pp. 280–291.
Wang, Y., Mei, J., Zhang, L., Zhang, B., Zhu, P., Li, Y., Li, X., 2018b. Self-supervised feature learning with crf embedding for hyperspectral image classification. IEEE Trans. Geosci. Remote Sens.
Waske, B., van der Linden, S., Benediktsson, J.A., Rabe, A., Hostert, P., 2010. Sensitivity of support vector machines to random feature selection in classification of hyperspectral data. IEEE Trans. Geosci. Remote Sens. 48 (7), 2880–2889.
Wei, D., Zhou, B., Torralba, A., Freeman, W.T., 2017a. mneuron: A Matlab Plugin to Visualize Neurons From Deep Models. Institute of Technology, Massachusetts.
Wei, W., Zhang, L., Tian, C., Plaza, A., Zhang, Y., 2017b. Structured sparse coding-based hyperspectral imagery denoising with intracluster filtering. IEEE Trans. Geosci. Remote Sens. 55 (12), 6860–6876.
Williams, R.J., Zipser, D., 1989. A learning algorithm for continually running fully recurrent neural networks. Neural Comput. 1 (2), 270–280.
Williams, T., Li, R., 2018. Wavelet pooling for convolutional neural networks. In: International Conference on Learning Representations, . <https://openreview.net/forum?id=rkhlb8lCZ>.
Windrim, L., Melkumyan, A., Murphy, R.J., Chlingaryan, A., Ramakrishnan, R., 2018. Pretraining for hyperspectral convolutional neural network classification. IEEE Trans. Geosci. Remote Sens. 56 (5), 2798–2810.
Wold, S., Esbensen, K., Geladi, P., 1987. Principal Component Analysis. Chemometrics Intell. Laborat. Syst. 2 (1), 37–52.
Wu, H., Prasad, S., 2017. Convolutional recurrent neural networks for hyperspectral data classification. Remote Sens. 9 (3), 298.
Wu, H., Prasad, S., 2018. Semi-supervised deep learning using pseudo labels for hyperspectral image classification. IEEE Trans. Image Process. 27 (3), 1259–1270.
Wu, Y., He, K., 2018. Group normalization. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 3–19.
Wu, Z., Li, Y., Plaza, A., Li, J., Xiao, F., Wei, Z., 2016. Parallel and distributed dimensionality reduction of hyperspectral data on cloud computing architectures. IEEE J. Sel. Top. Appl. Earth Obser. Remote Sens. 9 (6), 2270–2278.
Wyatte, D., Herd, S., Mingus, B., O'Reilly, R., 2012. The role of competitive inhibition and top-down feedback in binding during object recognition. Front. Psychol. 3, 182.
Xiaoli Jiao, C.-I.C., 2007. Unsupervised hyperspectral image classification. Imaging Spectrometry XII, vol. 6661https://doi.org/10.1117/12.732614. pp. 6661 – 6661 – 10.
Xie, J., Girshick, R., Farhadi, A., 2016. Unsupervised deep embedding for clustering analysis. In: International Conference on Machine Learning, pp. 478–487.
Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K., 2017. Aggregated residual transformations for deep neural networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5987–5995.
Xie, W., Li, Y., Jia, X., 2018. Deep convolutional networks with residual learning for accurate spectral-spatial denoising. Neurocomputing 312, 372–381.
Xing, C., Ma, L., Yang, X., 2016. Stacked Denoise Autoencoder Based Feature Extraction and Classification for Hyperspectral Images. J. Sensors 2016.
Xu, B., Gong, P., 2008. Noise estimation in a noise-adjusted principal component transformation and hyperspectral image restoration. Can. J. Remote Sens. 34 (3), 271–286.
Xu, B., Wang, N., Chen, T., Li, M., 2015a. Empirical evaluation of rectified activations in convolutional network. arXiv preprint arXiv:1505.00853.
Xu, X., Lil, f., Plaza, A., 2016a. Fusion of hyperspectral and LiDAR data using morphological component analysis. In: 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 3575–3578.
Xu, Y., Du, B., Zhang, F., Zhang, L., 2018. Hyperspectral image classification via a random patches network. ISPRS J. Photogramm. Remote Sens. 142, 344–357.
Xu, Y., Du, J., Dai, L., Lee, C., 2015b. A regression approach to speech enhancement based on deep neural networks. IEEE/ACM Trans. Audio, Speech, Language Process. 23 (1), 7–19.
Xu, Y., Wu, Z., Li, J., Plaza, A., Wei, Z., 2016. Anomaly detection in hyperspectral images based on low-rank and sparse representation. IEEE Trans. Geosci. Remote Sens. 54 (4), 1990–2000.
Yang, C., Everitt, J.H., Bradford, J.M., Murden, D., 2004. Airborne hyperspectral imagery and yield monitor data for mapping cotton yield variability. Precision Agric. 5 (5), 445–461.
Yang, H., 1999. A back-propagation neural network for mineralogical mapping from aviris data. Int. J. Remote Sens. 20 (1), 97–110.
Yang, J., Zhao, Y., Chan, J.C., Yi, C., 2016. Hyperspectral image classification using two-channel deep convolutional neural network. In: 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 5079–5082.
Yang, J., Zhao, Y.-Q., Chan, J.C.-W., 2017. Learning and transferring deep joint spectral–spatial features for hyperspectral classification. IEEE Trans. Geosci. Remote Sens. 55 (8), 4729–4742.
Yang, S., Jin, H., Wang, M., Ren, Y., Jiao, L., 2014. Data-driven compressive sampling and learning sparse coding for hyperspectral image classification. IEEE Geosci. Remote Sens. Lett. 11 (2), 479–483.
Yang, X., Ye, Y., Li, X., Lau, R.Y.K., Zhang, X., Huang, X., 2018. Hyperspectral image classification with deep learning models. IEEE Trans. Geosci. Remote Sens. 56 (9), 5408–5423.
Yi, C., Zhao, Y.Q., Chan, J.C.W., 2018. Hyperspectral image super-resolution based on spatial and spectral correlation fusion. IEEE Trans. Geosci. Remote Sens. 56 (7), 4165–4177.
Yi, C., Zhao, Y.Q., Yang, J., Chan, J.C.W., Kong, S.G., 2017. Joint hyperspectral super-resolution and unmixing with interactive feedback. IEEE Trans. Geosci. Remote Sens. 55 (7), 3823–3834.
Yosinski, J., Clune, J., Bengio, Y., Lipson, H., 2014. How transferable are features in deep neural networks? In: Advances in Neural Information Processing Systems. pp. 3320–3328.
Younos, T., Parece, T., 2015. Advances in Watershed Science and Assessment. The Handbook of Environmental Chemistry. Springer International Publishing.
Yu, D., Seltzer, M.L., Li, J., Huang, J., Seide, F., 2013. Feature learning in deep neural networks - A study on speech recognition tasks. CoRR abs/1301.3605.
Yu, D., Wang, H., Chen, P., Wei, Z., 2014. Mixed pooling for convolutional neural networks. In: Miao, D., Pedrycz, W., Śl?zak, D., Peters, G., Hu, Q., Wang, R. (Eds.), Rough Sets and Knowledge Technology. Springer International Publishing, Cham, pp. 364–375.
Yu, S., Jia, S., Xu, C., 2017. Convolutional neural networks for hyperspectral image classification. Neurocomputing 219, 88–98.
Yuan, Q., Zhang, Q., Li, J., Shen, H., Zhang, L., 2019. Hyperspectral image denoising employing a spatial–spectral deep residual convolutional neural network. IEEE Trans. Geosci. Remote Sens. 57 (2), 1205–1218.
Yue, J., Mao, S., Li, M., 2016. A deep learning framework for hyperspectral image classification using spatial pyramid pooling. Remote Sens. Lett. 7 (9), 875–884.
Yue, J., Zhao, W., Mao, S., Liu, H., 2015. Spectral-spatial classification of hyperspectral images using deep convolutional neural networks. Remote Sens. Lett. 6 (6), 468–477.
Zabalza, J., Ren, J., Zheng, J., Zhao, H., Qing, C., Yang, Z., Du, P., Marshall, S., 2016. Novel segmented stacked autoencoder for effective dimensionality reduction and feature extraction in hyperspectral imaging. Neurocomputing 185, 1–10.
Zeiler, M.D., 2012. Adadelta: an adaptive learning rate method. arXiv preprint arXiv:1212.5701.
Zeiler, M.D., Fergus, R., 2013. Stochastic pooling for regularization of deep convolutional neural networks. CoRR abs/1301.3557. URL http://arxiv.org/abs/1301.3557.
Zhan, Y., Hu, D., Wang, Y., Yu, X., 2018. Semisupervised hyperspectral image classification based on generative adversarial networks. IEEE Geosci. Remote Sens. Lett. 15 (2), 212–216.
Zhang, D., Kang, J., Xun, L., Huang, Y., 2019a. Hyperspectral image classification using spatial and edge features based on deep learning. Int. J. Pattern Recognit Artif Intell.
Zhang, F., Du, B., Zhang, L., Zhang, L., 2016a. Hierarchical feature learning with dropout k-means for hyperspectral image classification. Neurocomputing 187, 75–82 recent Developments on Deep Big Vision.
Zhang, H., He, W., Zhang, L., Shen, H., Yuan, Q., 2014. Hyperspectral image restoration using low-rank matrix recovery. IEEE Trans. Geosci. Remote Sens. 52 (8), 4729–4743.
Zhang, H., Li, Y., 2016. Spectral-spatial classification of hyperspectral imagery based on deep convolutional network. In: 2016 International Conference on Orange Technologies (ICOT), pp. 44–47.
Zhang, H., Li, Y., Jiang, Y., Wang, P., Shen, Q., Shen, C., 2019. Hyperspectral

classification based on lightweight 3-d-cnn with transfer learning. IEEE Trans. Geosci. Remote Sens.
Zhang, H., Li, Y., Zhang, Y., Shen, Q., 2017. Spectral-spatial classification of hyperspectral imagery using a dual-channel convolutional neural network. Remote Sens. Lett. 8 (5), 438–447.
Zhang, L., Du, B., 2012. Recent advances in hyperspectral image processing. Geo-spatial Informat. Sci. 15 (3), 143–156.
Zhang, L., Zhang, L., Du, B., 2016b. Deep learning for remote sensing data: A technical tutorial on the state of the art. IEEE Geosci. Remote Sens. Mag. 4 (2), 22–40.
Zhang, L., Zhang, L., Tao, D., Huang, X., 2012. On combining multiple features for hyperspectral remote sensing image classification. IEEE Trans. Geosci. Remote Sens. 50 (3), 879–893.
Zhang, M., Gong, M., Mao, Y., Li, J., Wu, Y., 2018a. Unsupervised feature extraction in hyperspectral images based on wasserstein generative adversarial network. IEEE Trans. Geosci. Remote Sens. 57 (5), 2669–2688.
Zhang, P., Gong, M., Su, L., Liu, J., Li, Z., 2016c. Change detection based on deep feature representation and mapping transformation for multi-spatial-resolution remote sensing images. ISPRS J. Photogramm. Remote Sens. 116, 24–41.
Zhang, X., Sun, Y., Jiang, K., Li, C., Jiao, L., Zhou, H., 2018b. Spatial sequential recurrent neural network for hyperspectral image classification. IEEE J. Sel. Top. Appl. Earth Obser. Remote Sens. 1–15.
Zhao, C., Wan, X., Zhao, G., Cui, B., Liu, W., Qi, B., 2017a. Spectral-spatial classification of hyperspectral imagery based on stacked sparse autoencoder and random forest. Eur. J. Remote Sens. 50 (1), 47–63.
Zhao, R., Luk, W., Niu, X., Shi, H., Wang, H., 2017b. Hardware acceleration for machine learning. In: 2017 IEEE Computer Society Annual Symposium on VLSI (ISVLSI). IEEE, pp. 645–650.
Zhao, W., Du, S., 2016. Spectral-spatial feature extraction for hyperspectral image classification: A dimension reduction and deep learning approach. IEEE Trans. Geosci. Remote Sens. 54 (8), 4544–4554.
Zhao, W., Guo, Z., Yue, J., Zhang, X., Luo, L., 2015. On combining multiscale deep learning features for the classification of hyperspectral remote sensing imagery. Int. J. Remote Sens. 36 (13), 3368–3379.
Zheng, Z., Zhang, Y., Li, L., Zhu, M., He, Y., Li, M., Guo, Z., He, Y., Yu, Z., Yang, X., Liu, X., Luo, J., Yang, T., Liu, Y., Li, J., 2017. Classification based on deep convolutional neural networks with hyperspectral image. In: 2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 1828–1831.
Zhong, P., Gong, Z., Li, S., Schönlieb, C.B., 2017a. Learning to diversify deep belief networks for hyperspectral image classification. IEEE Trans. Geosci. Remote Sens. 55 (6), 3516–3530.
Zhong, Y., Wang, X., Zhao, L., Feng, R., Zhang, L., Xu, Y., 2016a. Blind spectral unmixing based on sparse component analysis for hyperspectral remote sensing imagery. ISPRS J. Photogramm. Remote Sens. 119, 49–63.
Zhong, Z., Li, J., Luo, Z., Chapman, M., 2017b. Spectral-Spatial Residual Network for Hyperspectral Image Classification: A 3-D Deep Learning Framework. IEEE Trans. Geosci. Remote Sens. PP (99), 1–12.
Zhong, Z., Li, J., Ma, L., Jiang, H., Zhao, H., July 2017c. Deep residual networks for hyperspectral image classification. In: 2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS). pp. 1824–1827.
Zhou, F., Hang, R., Liu, Q., Yuan, X., 2018a. Hyperspectral image classification using spectral-spatial lstms. Neurocomputing.
Zhou, F., Hang, R., Liu, Q., Yuan, X., 2018b. Integrating convolutional neural network and gated recurrent unit for hyperspectral image spectral-spatial classification. In: Chinese Conference on Pattern Recognition and Computer Vision (PRCV). Springer, pp. 409–420.
Zhou, J., Liang, J., Qian, Y., Gao, Y., Tong, L., 2015. On the sampling strategies for evaluation of joint spectral-spatial information based classifiers. In: 2015 7th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS). IEEE, pp. 1–4.
Zhou, P., Han, J., Cheng, G., Zhang, B., 2019a. Learning compact and discriminative stacked autoencoder for hyperspectral image classification. IEEE Trans. Geosci. Remote Sens.
Zhou, S., Xue, Z., Du, P., 2019b. Semisupervised stacked autoencoder with cotraining for hyperspectral image classification. IEEE Trans. Geosci. Remote Sens.
Zhou, X., Li, S., Tang, F., Qin, K., Hu, S., Liu, S., 2017. Deep learning with grouped features for spatial spectral classification of hyperspectral images. IEEE Geosci. Remote Sens. Lett. 14 (1), 97–101.
Zhou, X.M., Wang, N., Wu, H., Tang, B.H., Li, Z.L., 2011. Estimation of precipitable water from the thermal infrared hyperspectral data. In: 2011 IEEE International Geoscience and Remote Sensing Symposium, pp. 3241–3244.
Zhu, J., Fang, L., Ghamisi, P., 2018a. Deformable convolutional neural networks for hyperspectral image classification. IEEE Geosci. Remote Sens. Lett. 15 (8), 1254–1258.
Zhu, J., Wu, L., Hao, H., Song, X., Lu, Y., June 2017a. Auto-encoder based for high spectral dimensional data classification and visualization. In: 2017 IEEE Second International Conference on Data Science in Cyberspace (DSC). pp. 350–354.
Zhu, L., Chen, Y., Ghamisi, P., Benediktsson, J.A., 2018b. Generative adversarial networks for hyperspectral image classification. IEEE Trans. Geosci. Remote Sens. 56 (9), 5046–5063.
Zhu, X.X., Tuia, D., Mou, L., Xia, G.-S., Zhang, L., Xu, F., Fraundorfer, F., 2017. Deep learning in remote sensing: A comprehensive review and list of resources. IEEE Geosci. Remote Sens. Mag. 5 (4), 8–36.
Zuo, Z., Shuai, B., Wang, G., Liu, X., Wang, X., Wang, B., Chen, Y., 2015. Convolutional recurrent neural networks: Learning spatial dependencies for image representation. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 18–26.

Escuela Politecnica
Av. de la Universidad, S/N, 10003
Caceres, Spain
Phone: 0034927257000. Ext. 51662
Email: aplaza,jplaza{@unex.es}

Dr. Antonio Plaza Miguel y Dr. Javier Plaza Miguel como directores de la tesis titulada "Procesamiento eficiente y profundo de imgenes hiperespectrales de la observación remota de la Tierra y aplicaciones en tareas de clasificación", certifico el factor de impacto y la categorización de la siguiente publicación, incluida en la tesis doctoral. Del mismo modo, se especifica la aportación del doctorado. Si necesitas cualquier información o clarificación, por favor, no dude en contactar conmigo.

Antonio Plaza Miguel PhD. and Javier Plaza Miguel PhD as directors of the Phd thesis titled "Deep-efficient processing of remote sensing hyperspectral images and applications in classification tasks.", certify the impact factor and the categorization of the following publication, included in the doctoral thesis. In the same way, the contribution of the doctorate is specified. If you need any further information or clarification, please do not hesitate contacting me.

Artculo / Paper

Autores/Authors: **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza.
Title: Scalable Recurrent Neural Network for Hyperspectral Image Classification.
Journal: Journal of Supercomputing.
Other Information: In press, 2020.
DOI: 10.1007/s11227-020-03187-0.
Impact factor 2018: 2.157. Q2
Abstract: Hyperspectral imaging (HSI) collects hundreds of images over large spatial observation areas on the Earths surface, recording scenes at different wavelength channels and providing a vast amount of information. Recurrent neural networks (RNNs) have been widely used for the classification of HSI datasets, understood as a single sequence of pixel vectors with high dimensionality. However, the RNN model scales poorly when dealing with HSI scenes with large dimensionality. In order to mitigate this problem, this paper presents a new RNN classifier based on simple recurrent units that performs HSI classification in a highly scalable and efficient way. Our experimental results (conducted on four real HSI datasets) reveal very good performance, not only in terms of classification accuracy (in line with existing methods), but also in terms of computational performance when dealing with large datasets.

Contribución del doctorado: Planteamiento de la hipótesis, desarrollo práctico, análisis y discusión de los resultados, elaboración y escritura del manuscrito.

Firma / Signature
Mayo / May, 2020

Antonio Plaza Miguel

Javier Plaza Miguel

The Journal of Supercomputing
https://doi.org/10.1007/s11227-020-03187-0

Scalable recurrent neural network for hyperspectral image classification

Mercedes E. Paoletti[1] · **Juan M. Haut**[1] · **Javier Plaza**[1] · **Antonio Plaza**[1]

Abstract

Hyperspectral imaging (HSI) collects hundreds of images over large spatial observation areas on the Earth's surface, recording scenes at different wavelength channels and providing a vast amount of information. Recurrent neural networks (RNNs) have been widely used for the classification of HSI datasets, understood as a single sequence of pixel vectors with high dimensionality. However, the RNN model scales poorly when dealing with HSI scenes with large dimensionality. In order to mitigate this problem, this paper presents a new RNN classifier based on simple recurrent units that performs HSI classification in a highly scalable and efficient way. Our experimental results (conducted on four real HSI datasets) reveal very good performance, not only in terms of classification accuracy (in line with existing methods), but also in terms of computational performance when dealing with large datasets.

Keywords Hyperspectral image · Recurrent neural networks · CUDA

1 Introduction

The significant advances in computing technology achieved in the last decade, coupled with the newest developments in imaging spectroscopy [18], have allowed the development of new Earth observation (EO) missions with powerful airborne and satellite hyperspectral imaging (HSI) sensors, which can capture high-quality

Source codes: https://github.com/mhaut/scalable_RNN_HSI.

✉ Juan M. Haut
juanmariohaut@unex.es

Mercedes E. Paoletti
mpaoletti@aunex.es

Javier Plaza
jplaza@unex.es; aplaza@unex.es

[1] Department of Technology of Computers and Communications, University of Extremadura, Escuela Politecnica, Avda. de la Universidad s/n, Cáceres, Spain

Published online: 04 February 2020

images composed by hundreds of measurements (at different wavelength channels) over extensive spatial areas, acquiring information in hundreds of continuous and narrow bands, ranging from the visible to the near-infrared (NIR) and shortwave-infrared (SWIR) [8] parts of the electromagnetic spectrum.

As a result, current spectrometers are able to produce very large HSI data cubes, where each pixel contains the spectral signature of the observed materials. These spectral signatures collect the physical–chemical behavior of materials in the presence of solar light, being unique for each kind of terrestrial object, and allowing to describe and identify each element of the scene, not only at an object level, but also at pixel (and even sub-pixel) level of detail [19], providing abundant information for the characterization of the surface of the Earth. Such information can be used in a wide range of human activities, such as hydrology [12], forestry [20], geology [22] and mineralogy [5]), as well as precision agriculture [7], urban planning [27], and prevention and management of disasters.

In this context, the analysis and processing of HSI datasets plays an important role in remote sensing [14], demanding the development of effective and efficient techniques for the analysis of these data. In particular, this paper focuses on methods that identify the content of an HSI scene by associating each pixel in the scene with a corresponding land-cover class. These methods can be described as the mapping function $f(\cdot, \theta)$, where $f : \mathbf{X} \rightarrow \mathbf{Y}$ is able to relate each pixel of the HSI scene $\mathbf{x}_i \in \mathbf{X}$ with a land-cover category $\mathbf{y}_i$ by adjusting its parameters θ, obtaining at the end a classification map $\mathbf{Y}$ with pairs of the form: $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{n_{\text{pixels}}}$.

Besides, it must be noted that HSI data present several challenges related to its volume and complexity. While a detailed spectral signature allows for the unique identification of each material in the scene, its large spectral dimension and content can further complicate the classification process, imposing significant storage and processing restrictions. As a result, classification methods must be efficient and scalable in the use of computational and storage elements [25]. Moreover, the use of a large number of spectral features increases the complexity of classification methods, hampering their performance and leading to lower classification accuracies with more features (*peaking paradox*). In the field of HSI, this issue corresponds with the *curse of dimensionality* phenomenon that, coupled with high intraclass variability and interclass similarity, makes the classification problem an extremely ill-posed one.

Among available classification methods, deep neural networks (DNNs) [3] have attracted the attention of the HSI research community due to several characteristics that facilitate the classification of these kinds of data, including the following aspects:

- They do not need prior information about statistical properties of the HSI data to extract and process the spectral, spatial and spectral–spatial information contained in the scenes.
- Their working mode is based on the optimization of a loss function, for instance the mean square error (MSE) between the networks' outputs and the desired outputs, through the adjustment of the networks' parameters θ. To achieve this, they employ a forward-backward iterative mechanism (based on gradient descent

optimizers) to find the optimal θ, being able to work as universal approximators [6].
- They offer great flexibility in terms of learning models, i.e., unsupervised, supervised and semi-supervised.
- As stacks of layers composed by neurons, they provide a great variety of architectures, from shallow to deep and very deep ones, employing fully (or locally) connected neurons, and implementing one or more paths.

Moreover, advances in computing technology have allowed the implementation of deepest and more complex neural models, which have led to a revolution in deep learning (DL) techniques [15]. Focusing on HSI classification, these DNNs are able to extract representative features, learning simple representations at the first layers and extending them to more complex abstractions at the final layers (hierarchical learning), discovering nonlinear relationships in the input HSI data and yielding high performance in HSI classification [24, 32].

In particular, the recurrent neural network (RNN) [29] is an interesting classifier that presents an internal structure similar to a directed graph, implementing loops in the connections of the layers that force each node activation of the current step to depend on the activations of the preceding ones. In this sense, the RNN is a powerful model for learning sequences of data, storing an internal state that provides a memory to relate the current input data sample with the previous ones. At the end, these states allow to process the contextual information of the data, extracting temporal features. The RNN has been previously employed to perform HSI data classification, considering each spectral pixel as the input sequential data of the model, as Mou et al. propose in [21]. Other approaches even combine spectral information with spatial information. For instance, Zhou et al. [34] concatenate the spatial information of a neighborhood window (extracted with PCA [30]) to the spectral information of the pixel. Zhang et al. [33] consider several principal components (for which they extract Gabor textures and differential morphological profiles) which are combined and stacked to conform local spatial sequential (LSS) features that will be sent as input to the RNN model.

Nonetheless, all the aforementioned spectral and spectral–spatial methodologies must face an important restriction. In particular, RNNs have been shown to suffer from overfitting when the length of their input sequences is very large, which happens often when dealing with HSIs, since spectral bands are handled as sequences of length n_3 in which each sequence feature is a band, so overfitting becomes more evident as n_3 increases. To mitigate this problem, it is usual to include more information to the model, e.g., grouping the spectral bands to enlarge the size of the features and shorten the length of the sequence. However, this practice hampers the scalability and performance of the network. Also, as the sequence and/or feature length increases, the runtimes needed by the model become longer.

In order to improve the performance of RNN models (in terms of both runtimes and scalability), some interesting parallel versions and toolkits have been developed [1, 17, 23]. For instance, [28] presents a general RNN with a parallel implementation for graphics processing units (GPUs) based on NVIDIA CUDA. Other works focus on the skip of the hidden states [2] in order to speed up the data processing.

However, to the best of our knowledge, very few efforts have been focused on accelerating the processing of HSIs with RNNs. However, accelerating RNNs for HSI data processing can give an adequate solution to the problems of scalability, runtime and overfitting when dealing with the high spectral dimensionality of these kinds of data.

This paper investigates (for the first time in the available literature) the scalable implementation of a novel variant of the RNN model, called simple recurrent unit (SRU) [16], for HSI data classification. Comparing the SRU with traditional recurrent units, its architecture allows faster learning in terms of training speed, reducing the number of trainable parameters while maintaining a reliable performance (in terms of accuracy). Thus, the main goal of this paper is to reduce the internal complexity of the RNN model (i.e., the relationships between the current outputs and the previous ones), thus facilitating the parallelization of the computations performed by the recurrent units in order to enhance the performance of traditional RNN models for HSI data classification.

2 Recurrent neural units: overview

An HSI data scene $\mathbf{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ can be represented as a matrix of $n_1 \times n_2$ pixels, where each pixel $\mathbf{x}_i$ is composed by n_3 spectral bands. On this wise, the classification pursues to associate each pixel with a corresponding land-cover class (label), obtaining a classification map $\mathbf{Y} \in \mathbb{R}^{n_1 \times n_2 \times n_{\text{classes}}} \equiv \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{n_1 \cdot n_2}$.

RNN models for HSI data interpretation process the spectral signature contained in each $\mathbf{x}_i$ as a time sequence, where the spectral bands are considered as time steps (see Fig. 1). This allows the pixels to be processed in band-by-band fashion [21]. Alternatively, the bands can be arranged into groups [31]. Moreover, the RNN model provides three different units: (i) vanilla recurrent unit, (ii) long short-term memory (LSTM) [11], and (iii) gated recurrent unit (GRU) [4]. The vanilla unit is the oldest and simplest model, as defined by Eq. (1):

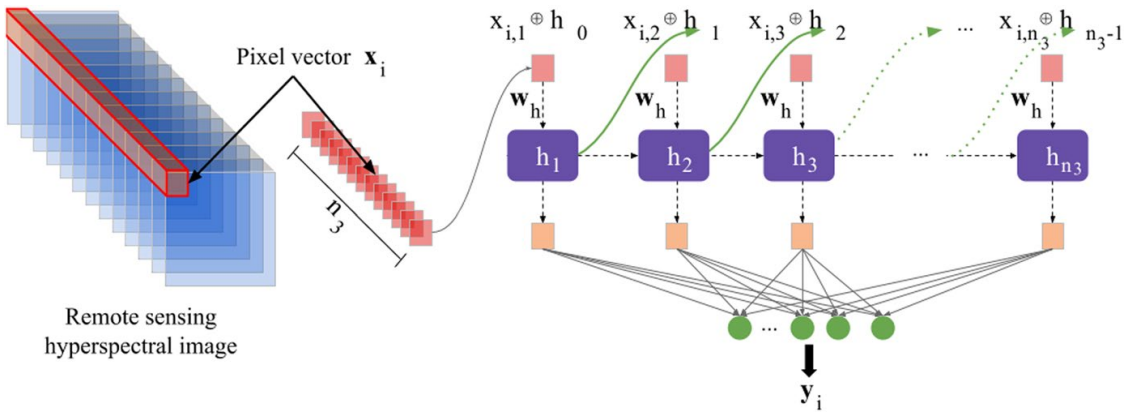


Fig. 1 Traditional band-by-band RNN model for HSI data classification. The spectral signature contained in each pixel is processed as a temporal sequence, where different spectral bands correspond to different time steps

$$\mathbf{h}_t = \begin{cases} 0 & \text{if } t = 0 \\ \mathcal{H}\left(\mathbf{W}_h \cdot \mathbf{x}_t + \mathbf{U}_h \cdot \mathbf{h}_{t-1} + b_h\right) & \text{if } t \neq 0 \end{cases} \tag{1}$$

Considering $\mathbf{x}_t \in \mathbb{R}^n$ as the input sequential sample, the vanilla model computes the output $\mathbf{y}_t$ as a hidden state at time t, i.e., $\mathbf{y}_t = \mathbf{h}_t$, by combining the current input with the unit's data weights $\mathbf{W}_h$ and bias b_h and the previous state $\mathbf{h}_{t-1}$, weighted by the connection weights $\mathbf{U}_h$, being $\mathcal{H}(\cdot)$ a nonlinear activation function such as sigmoid or hyperbolic tangent (tanh). As a result, the obtained hidden state works as the memory of the model, and it is applied to the subsequent sample $\mathbf{x}_{t+1}$.

Due to its simplicity, the vanilla RNN tends to quickly degrade the interpretation of high-dimensional data, reaching poor classification results. The LSTM deals with the data degradation by reinterpreting the original RNN as a cell composed by two main states, the hidden $\mathbf{h}_t$ and cell $\mathbf{c}_t$ states, which are controlled by three gates known as input $\mathbf{i}_t$, output $\mathbf{o}_t$ and forget $\mathbf{f}_t$ gates, in order to manage the information flow that goes through the unit. This mechanism allows the LSTM to learn useful information along time, disregarding the irrelevant one.

$$\begin{aligned} \mathbf{i}_t &= \mathcal{H}\left(\mathbf{W}_i \cdot \mathbf{x}_t + \mathbf{U}_i \cdot \mathbf{h}_{t-1} + b_i\right) \\ \mathbf{f}_t &= \mathcal{H}\left(\mathbf{W}_f \cdot \mathbf{x}_t + \mathbf{U}_f \cdot \mathbf{h}_{t-1} + b_f\right) \\ \mathbf{o}_t &= \mathcal{H}\left(\mathbf{W}_o \cdot \mathbf{x}_t + \mathbf{U}_o \cdot \mathbf{h}_{t-1} + b_o\right) \\ \mathbf{c}_t &= \begin{cases} 0 & \text{if } t = 0 \\ \mathbf{f}_t \circ \mathbf{c}_{t-1} + \mathbf{i}_t \circ \mathcal{H}\left(\mathbf{W}_c \cdot \mathbf{x}_t + \mathbf{U}_c \cdot \mathbf{h}_{t-1} + b_c\right) & \text{if } t \neq 0 \end{cases} \\ \mathbf{h}_t &= \begin{cases} 0 & \text{if } t = 0 \\ \mathbf{o}_t \circ \mathcal{H}\left(\mathbf{c}_t\right) & \text{if } t \neq 0 \end{cases} \end{aligned} \tag{2}$$

As we can observe in Eq. (2), $\mathbf{h}_t$ works as the traditional output of the unit, while $\mathbf{c}_t$ includes (or removes) information into (from) the cell, depending on the gates' values. Thus, the input gate $\mathbf{i}_t$ determines whether a new input sample is allowed to reach inside the cell or not; the forget gate $\mathbf{f}_t$ deletes the irrelevant information; and the output gate $\mathbf{o}_t$ weights the unit's output signal at time t.

However, this control-gate mechanism introduces significant complexity to LSTM. With this in mind, the GRU model tries to make a compromise between the simplicity of the vanilla unit and the high performance of the LSTM. In fact, the GRU can be considered as a simplified LSTM, with the output gate removed (this involves fewer parameters) and the input and forget gates evolved into update ($\mathbf{z}_t$) and reset ($\mathbf{r}_t$) gates, as Eq. (3) shows:

$$\begin{aligned} \mathbf{z}_t &= \mathcal{H}\left(\mathbf{W}_z \cdot \mathbf{x}_t + \mathbf{U}_z \cdot \mathbf{h}_{t-1} + b_z\right) \\ \mathbf{r}_t &= \mathcal{H}\left(\mathbf{W}_r \cdot \mathbf{x}_t + \mathbf{U}_r \cdot \mathbf{h}_{t-1} + b_r\right) \\ \mathbf{h}'_t &= \tanh\left(\mathbf{W}_h \cdot \mathbf{x}_t + \mathbf{r}_t \circ \mathbf{U}_h \cdot \mathbf{h}_{t-1} + b_h\right) \\ \mathbf{h}_t &= \begin{cases} 0 & \text{if } t = 0 \\ \mathbf{z}_t \circ \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \circ \mathbf{h}'_t & \text{if } t \neq 0 \end{cases} \end{aligned} \tag{3}$$

Although these models are able to reach acceptable performance in HSI data classification, they present an important computational restriction due to the high dependence on previous steps. In fact, although algebraic operations can be optimized and parallelized in hardware, such dependencies hamper the speedup as the amount of data (and the dimensionality of the feature space) grow, making the RNN model scale poorly in this context. In addition, the RNN suffers from overfitting and vanishing gradient [10] problems. When processing HSI data, these models can see their performance severely compromised when the spectral dimension of the images is too high. In order to overcome these limitations, in the next section we introduce a new RNN-based architecture that employs a SRU as the main block of the model.

3 Proposed method

3.1 SRU methodology

The SRU simplifies the internal architecture of the recurrent cell. With a design in-between that of the LSTM and GRU models, it exhibits two main states: the hidden state $\mathbf{h}_t$ and the cell state $\mathbf{c}_t$, controlled by the forget $\mathbf{f}_t$ and reset $\mathbf{r}_t$ gates, as Eq. (4) indicates:

$$\begin{aligned} \mathbf{f}_t &= \mathcal{H}\left(\mathbf{W}_f \cdot \mathbf{x}_t + \mathbf{u}_f \odot \mathbf{c}_{t-1} + b_f\right) \\ \mathbf{r}_t &= \mathcal{H}\left(\mathbf{W}_r \cdot \mathbf{x}_t + \mathbf{u}_r \odot \mathbf{c}_{t-1} + b_r\right) \\ \mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \left(1 - \mathbf{f}_t\right) \odot \left(\mathbf{W}_c \cdot \mathbf{x}_t\right) \\ \mathbf{h}_t &= \mathbf{r}_t \odot \mathbf{c}_t + (1 - \mathbf{r}_t) \odot \mathbf{x}_t \cdot \alpha \end{aligned} \tag{4}$$

Internally, Eq. (4) can be divided in two main components: (i) the *light recurrence*, which reads the input $\mathbf{x}_t$ at time t and computes the forget gate $\mathbf{f}_t$ and the cell state $\mathbf{c}_t$ values (capturing the relevant sequential information), and (ii) the *highway network*, which obtains the unit's output by using the reset gate $\mathbf{r}_t$ and the hidden state $\mathbf{h}_t$.

Focusing on the light recurrence concept, we observe that gates and cell states are not obtained by relying on previous hidden states (i.e., $\mathbf{h}_{t-1}$), but on the previous cell state (i.e., $\mathbf{c}_{t-1}$). Furthermore, the SRU units do not integrate the previous states through a matrix multiplication ($\cdot$), which is a quite complex mathematical operation to parallelize at the hardware level. This is because each dimension of the resulting output depends on all the entries of $\mathbf{c}_{t-1}$, imposing a "waiting time" (in fact, a delay) until the state $\mathbf{c}_{t-1}$ is fully computed. In turn, the SRU performs a point-wise multiplication ($\odot$), allowing the independence of each output's dimension to obtain part of the output without having the previous state $\mathbf{c}_{t-1}$ fully calculated. This point-wise multiplication is also employed by the highway network [26], which adaptively combines the current input $\mathbf{x}_t$ with the cell state $\mathbf{c}_t$ through the reset gate $\mathbf{r}_t$. Moreover, it reduces the vanishing gradient problem by implementing a skip connection $(1 - \mathbf{r}_t) \odot \mathbf{x}_t \cdot \alpha$, controlled by the *scaling correction* constant α, which helps to propagate the gradient signal.

3.2 Proposed CUDA-based SRU for HSI classification

The proposed SRU has been implemented over a GPU using CUDA, aiming at increasing the model's performance by parallelizing the operations given in Eq. (4). In this context, it must be noted that the HSI data have been reshaped into an $n \times n_3$ matrix (with $n = n_1 \times n_2$) and divided into training $\mathcal{D}_{\text{train}}$ and test $\mathcal{D}_{test}$ subsets to adjust the parameters and validate the model, being $\mathcal{D}_{\text{train}}$ organized as batches $\mathbf{X} \in \mathbb{R}^{n_3 \times n_b}$, having each one n_b sequences (i.e., pixels) of n_3 features (i.e., spectral bands).

Besides, the network has been implemented as a *many-to-one* model with one cell that computes across input sequences to perform a single prediction from each one, defining the hidden space $\mathbb{R}^{n_h}$. From Eq. 4, each weight matrix $\mathbf{W}_*$ dotted by $\mathbf{X}$ is extracted, fusing these computations (by using cuBLAS) into a single matrix multiplication that obtains the auxiliary matrix $\mathbf{M} \in \mathbb{R}^{n_b \times k \cdot n_h}$ of Fig. 2. It is worth mentioning that, as a highway network's point-wise operation is performed, if the input and output data sizes are not the same, the $\mathbf{M}$ will introduce a fourth matrix that will serve as the highway connection's weights, setting $k = 4$ and performing $(1 - \mathbf{r}_t) \odot (\mathbf{W} \cdot \mathbf{x}_t)$ in Eq. (4). With this in mind, a CUDA kernel composed by $n_b \cdot n_h$ threads, arranged in the form of one-dimensional blocks, has been implemented. Algorithm 1 shows a pseudocode of the aforementioned kernel, where all the data structures (i.e., matrices and vectors) have been stored considering the C-style (i.e., row-major order) memory storage scheme. Moreover, Fig. 3 gives a graphical example of how the CUDA threads access the positions of matrix $\mathbf{M}$. Finally, our model

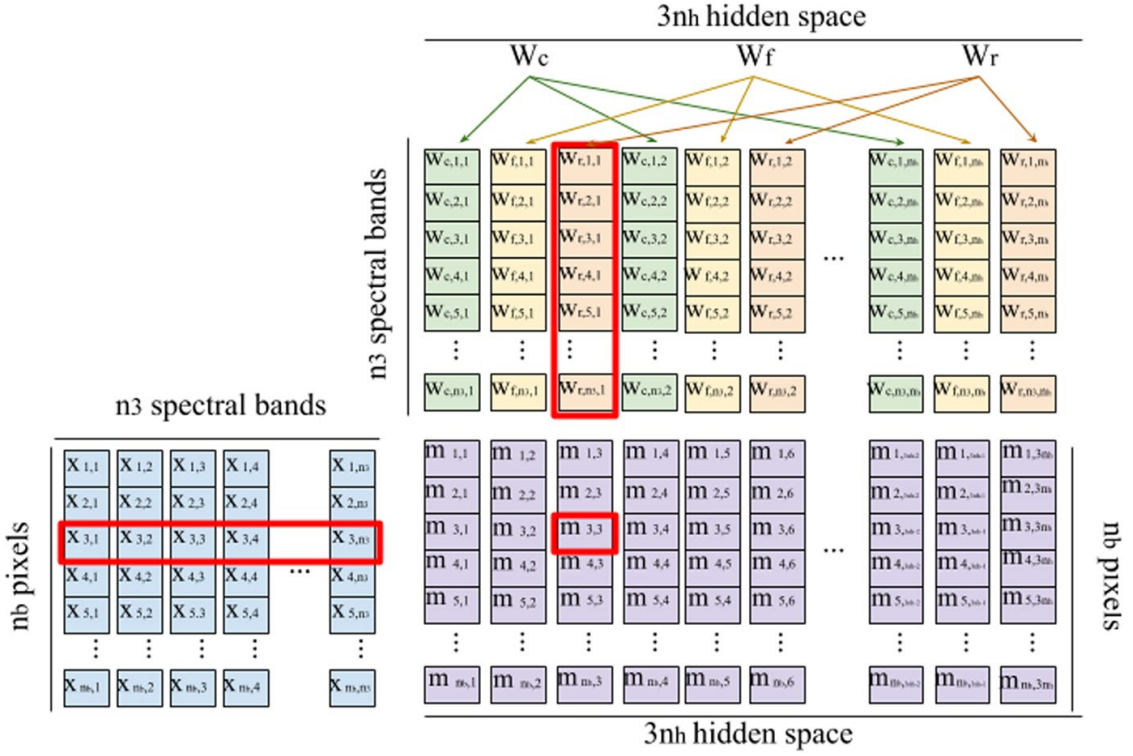


Fig. 2 General matrix-to-matrix multiplication between the current HSI data batch and the model's weights, optimized through cuBLAS library

Fig. 3 Diagram illustrating the threads' access to auxiliary matrix **M**, where black, red green and purple arrows represent those threads with identifier 0, 1, 2 and $n_b \cdot n_h - 1$, respectively

considers the following hyperparameters: $n_b = 100$, $n_h = 144$, $n_3 =$ spectral bands and $k = 4$, setting the maximum number of threads per block to 512.

Algorithm 1 Pseudocode of SRU model to perform HSI data classification

procedure HSI-SRU($\mathbf{X} \in \mathbb{R}^{n_b \times n_3}$: HSI data batch, $\mathbf{M} \in \mathbb{R}^{n_b \times k \cdot n_h}$: auxiliary matrix, $\mathbf{c}_{t-1} \in \mathbb{R}^{n_3 \times n_b \cdot n_h}$: previous cell state, $\mathbf{u} \in \mathbb{R}^{2n_h}$: weight vectors $\mathbf{u}_f$ and $\mathbf{u}_r$, $\mathbf{b} \in \mathbb{R}^{2n_h}$: biases vectors $\mathbf{b}_f$ and $\mathbf{b}_r$, n_b: batch size, n_h: hidden space size, n_3: spectral dimensions, k: factor)
  $i = blockIdx.x * blockDim.x + threadIdx.x$ ▷ current thread id
  if $i < (n_b \cdot n_h)$ **then** ▷ check that the maximum allowed dimension is not exceeded
    $N = n_b \cdot n_h \cdot k$ ▷ Elements of auxiliary matrix **M**
    $m = i \cdot k$ ▷ Index of auxiliary matrix **M**
    for $l = 0, \ldots, n_3$ **do** ▷ Go along the sequences
      $\mathbf{f}_t = \sigma\left(\mathbf{M}[m+1] + \mathbf{u}[i\%n_h] \cdot \mathbf{c}_{t-1}[i] + \mathbf{b}[(i\%n_h)]\right)$
      $\mathbf{r}_t = \sigma\left(\mathbf{M}[m+2] + \mathbf{u}[(i\%n_h) + n_h] \cdot \mathbf{c}_{t-1}[i] + \mathbf{b}[(i\%n_h)]\right)$
      $\mathbf{c}_t = \mathbf{f}_t \cdot \mathbf{c}_{t-1}[i] + (1 - \mathbf{f}_t) \cdot \mathbf{M}[m]$
      $\mathbf{h}_t = \mathbf{r}_t \cdot \mathbf{c}_t + (1 - \mathbf{r}_t) \cdot \mathbf{M}[m+3]$
      $m = m + N$ ▷ Updating index of **M**
      $\mathbf{c}_{t-1} = \mathbf{c}_t$ ▷ Updating previous state
      $\mathbf{c} = \text{Concatenate}(\mathbf{c}_t)$
      $\mathbf{h} = \text{Concatenate}(\mathbf{h}_t)$
    end for
  end if
  return h and **c**
end procedure

4 Experimental results

4.1 Hyperspectral datasets

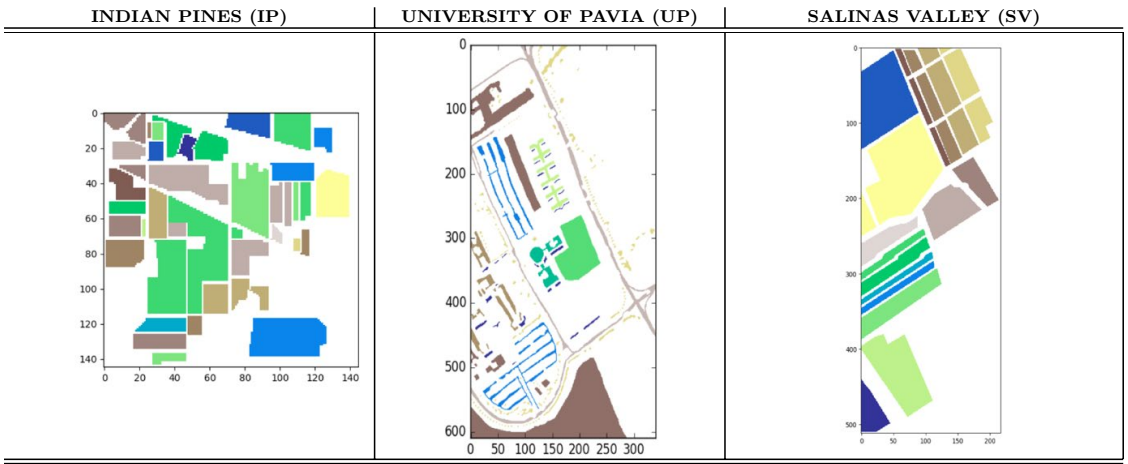
In order to test the proposed SRU model for HSI classification purposes, we perform an exhaustive comparison between all the available recurrent architectures: vanilla RNN, LSTM and GRU (implemented with CUDA and CuDNN) and the proposed method, with the aim of quantifying the computational improvements and the advantages that can be obtained in terms of performance and classification accuracy. To this end, four widely used images in the field of HSI data classification have been considered: Indian Pines (IP), Big Indian Pines (BIG) and Salinas Valley (SV), collected by AVIRIS, and the University of Pavia (UP) scene,

collected by ROSIS. Figure 4 shows a summary of these datasets, including the number of available labeled data in each case.

The IP, BIP and SV scenes were gathered by AVIRIS sensor [9] in 1992 over agricultural areas and comprise $145 \times 145 \times 200$, $2678 \times 614 \times 220$ and $512 \times 217 \times 204$ samples, respectively, organized in 16 and 58 different land-cover classes. The UP scene was captured by ROSIS [13] over an urban area, comprising $610 \times 340 \times 113$ samples labeled with 9 different classes.

4.2 Experimental environment

Several implementations of the considered RNN-based models for HSI classification have been developed and tested on a hardware environment with a i9-9940X processor, located over an Gigabyte X299 Aorus, 128GB of DDR4 RAM, and an NVIDIA Titan RTX GPU with 24GB GDDR6. In order to provide an efficient implementation, the proposed model (together with the different RNN architectures) has been parallelized on the GPU using CUDA 10.0.130 and cuDNN 7.6.0 language over the Pytorch framework, with Ubuntu 18.04.3 x64 as operating system.



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

Fig. 4 Number of available labeled samples in the Indian Pines (IP), University of Pavia (UP), and Salinas Valley (SV) HSI datasets

4.3 Experimental settings

Two main experiments have been conducted to perform an exhaustive analysis of the performance and scalability of the proposed method, as compared with other RNN architectures:

- Our first experiment compares the RNN models with a fixed percentage of training samples. In particular, 15% of randomly selected samples from the IP and BIP scenes and 10% from the UP and SV scenes have been considered. The RNN models have been implemented following the architecture proposed in [21]. The main goal is to study the classification performance of the methods, using standard metrics such as the overall (OA) and average (AA) accuracy, and the kappa coefficient (K). Also run times and number of parameters have been measured. Moreover, two different implementations of vanilla RNN, LSTM and GRU have been considered: (i) direct CUDA implementations and (ii) CuDNN library-based implementations.
- Our second experiment evaluates the scalability and speedup of the proposed model with different sizes of the input features. For this purpose, we have reduced the dimensionality of the scenes (while retaining most of the information in the scenes) using PCA. In particular, for the AVIRIS datasets, in addition to the original 200, 204 and 220 spectral bands, the RNN models have been tested with 50, 100 and 150 principal components while, for the UP scene, 10, 40 and 80 principal components have been, respectively, considered, in addition to the original 103 spectral bands.

4.4 Experimental discussion

4.4.1 Experiment 1: analysis of classification results

The results obtained in our first experiment are reported in Tables 1, 2 and 3. If we focus on the classification accuracy, we can observe that, in general, the CUDA implementations and their CuDNN counterparts reach similar OA, AA and K values, where vanilla RNN usually reaches the lowest accuracies. Comparing these results with those obtained by the SRU-based model, we can conclude that the SRU achieves intermediate results between the vanilla and LSTM/GRU, except for BIP and SV scenes, where it reaches slightly worse results than the vanilla model (the reason for this is that a strong overfitting was observed during the training stage, while the complexity of the BIP's classes is greater than in other datasets). This indicates that the simplicity of the SRU (compared with LSTM and GRU) can negatively affect the overfitting of the network (as it happens also with the vanilla RNN), requiring regularization and standardization mechanisms (such as dropout) in order to reduce this effect. The resulting classification maps can be seen in Fig. 5. As the considered methods are pixel based, they all exhibit some "salt & pepper" noise in the classification results being all very similar.

Table 1 Classification results obtained for the IP dataset, using 15% randomly selected samples to perform the training stage

Class	Vanilla RNN		LSTM		GRU		Proposed
	CUDA	CuDNN	CUDA	CuDNN	CUDA	CuDNN	
1	28.72 ± 7.5	22.05 ± 4.76	38.97 ± 13.51	43.59 ± 9.59	**55.9** ± 6.96	38.97 ± 14.27	35.9 ± 9.32
2	73.82 ± 2.15	70.7 ± 2.24	**78.19** ± 3.33	75.4 ± 3.28	76.57 ± 1.28	75.7 ± 3.34	75.8 ± 2.75
3	55.63 ± 3.86	54.52 ± 4.39	62.21 ± 4.25	**66.18** ± 2.5	64.62 ± 3.59	61.76 ± 2.75	65.11 ± 4.43
4	41.79 ± 7.41	43.28 ± 5.55	55.92 ± 6.51	**56.52** ± 9.64	55.42 ± 4.72	55.32 ± 7.18	54.23 ± 12.67
5	86.1 ± 2.7	84.98 ± 2.08	89.46 ± 3.36	89.46 ± 1.37	86.15 ± 4.23	**90.88** ± 2.83	86.49 ± 2.34
6	95.0 ± 3.03	94.26 ± 1.4	95.48 ± 1.17	96.42 ± 1.71	**97.0** ± 1.68	97.0 ± 0.8	96.97 ± 1.04
7	40.0 ± 12.72	33.91 ± 10.79	61.74 ± 16.36	63.48 ± 2.13	61.74 ± 6.96	66.09 ± 13.01	**68.7** ± 6.39
8	97.98 ± 1.06	97.83 ± 1.7	99.41 ± 0.51	99.46 ± 0.33	**99.8** ± 0.24	99.61 ± 0.25	98.87 ± 1.32
9	10.59 ± 6.86	15.29 ± 6.0	36.47 ± 9.41	25.88 ± 8.8	24.71 ± 14.6	**43.53** ± 17.69	29.41 ± 11.16
10	67.58 ± 2.72	70.0 ± 7.05	73.73 ± 2.76	**75.86** ± 3.33	73.34 ± 2.49	73.34 ± 3.21	74.55 ± 1.25
11	77.16 ± 1.04	76.66 ± 2.75	**82.17** ± 2.22	79.69 ± 1.28	81.11 ± 1.3	80.65 ± 2.74	79.37 ± 1.48
12	65.6 ± 3.94	67.5 ± 8.25	**81.35** ± 3.54	80.99 ± 0.93	75.4 ± 1.31	80.44 ± 5.54	75.28 ± 4.35
13	96.44 ± 3.28	97.24 ± 1.23	98.51 ± 1.07	98.74 ± 0.43	98.51 ± 0.46	98.28 ± 0.89	**99.2** ± 0.46
14	93.6 ± 1.52	94.55 ± 1.3	95.33 ± 2.57	94.98 ± 2.4	95.0 ± 0.98	**95.52** ± 1.38	94.85 ± 1.38
15	63.84 ± 4.24	59.21 ± 4.47	70.73 ± 6.24	70.85 ± 5.26	71.59 ± 3.77	**72.8** ± 2.82	65.18 ± 4.22
16	83.29 ± 4.41	85.06 ± 2.93	90.13 ± 4.03	90.13 ± 5.63	**91.65** ± 4.5	88.1 ± 5.47	88.61 ± 2.12
OA	76.74 ± 0.62	76.3 ± 0.73	**81.9** ± 0.59	81.47 ± 0.55	81.25 ± 0.2	81.33 ± 0.52	80.48 ± 0.47
AA	67.32 ± 1.05	66.69 ± 1.5	75.61 ± 2.33	75.48 ± 1.63	75.53 ± 1.04	**76.12** ± 2.41	74.28 ± 1.62
K(x100)	73.37 ± 0.74	72.86 ± 0.9	**79.29** ± 0.69	78.84 ± 0.63	78.55 ± 0.25	78.66 ± 0.6	77.68 ± 0.52
Parameters	234704		247568		243280		**230928**
Runtime(s)	177.5 ± 0.7	48.42 ± 0.77	170.26 ± 1.45	53.58 ± 1.37	175.9 ± 0.99	49.69 ± 1.54	**29.9** ± 0.23
Speedup	1.0	3.67	1.04	3.31	1.01	3.57	**5.94**

The best metric values are in bold

Table 2 Classification results obtained for the UP dataset, using 10% randomly selected samples to perform the training stage

Class	Vanilla RNN		LSTM		GRU		Proposed
	CUDA	CuDNN	CUDA	CuDNN	CUDA	CuDNN	
1	92.69 ± 1.79	91.47 ± 0.97	93.35 ± 1.09	**94.29** ± 0.93	94.04 ± 0.6	94.1 ± 1.12	93.67 ± 0.95
2	97.67 ± 0.44	97.22 ± 0.5	**97.99** ± 0.28	97.55 ± 0.43	97.92 ± 0.36	97.43 ± 0.54	97.56 ± 0.33
3	72.08 ± 4.46	72.9 ± 2.86	**78.73** ± 3.49	76.35 ± 1.71	77.81 ± 1.39	76.0 ± 3.18	76.83 ± 0.96
4	92.76 ± 1.5	93.46 ± 1.09	93.46 ± 0.64	94.86 ± 0.44	**94.89** ± 1.14	94.53 ± 1.51	93.95 ± 1.23
5	99.69 ± 0.06	99.6 ± 0.1	99.65 ± 0.19	99.8 ± 0.13	99.8 ± 0.15	**99.87** ± 0.11	99.62 ± 0.17
6	89.16 ± 0.85	**90.99** ± 1.39	89.32 ± 0.84	90.22 ± 1.78	89.21 ± 1.93	90.1 ± 1.86	89.38 ± 1.45
7	81.64 ± 7.1	**86.6** ± 5.41	86.52 ± 1.82	83.59 ± 5.03	85.63 ± 4.44	86.55 ± 2.58	83.76 ± 1.27
8	85.33 ± 3.13	85.46 ± 1.52	88.29 ± 1.41	88.03 ± 1.9	87.67 ± 2.26	**88.4** ± 2.19	87.2 ± 1.89
9	99.79 ± 0.11	**99.93** ± 0.06	99.69 ± 0.22	99.67 ± 0.38	99.6 ± 0.06	99.81 ± 0.16	99.86 ± 0.14
OA	92.84 ± 0.32	92.93 ± 0.09	93.88 ± 0.2	93.81 ± 0.06	**93.92** ± 0.06	93.81 ± 0.12	93.51 ± 0.1
AA	90.09 ± 0.98	90.85 ± 0.64	**91.89** ± 0.32	91.59 ± 0.45	91.84 ± 0.46	91.87 ± 0.5	91.31 ± 0.23
K(x100)	90.48 ± 0.41	90.61 ± 0.12	91.86 ± 0.26	91.78 ± 0.08	**91.92** ± 0.09	91.77 ± 0.17	91.38 ± 0.13
Parameters	76809		89673		85385		**73033**
Runtime(s)	351.9 ± 1.33	109.47 ± 0.91	303.16 ± 2.02	118.22 ± 0.61	295.75 ± 1.19	110.6 ± 1.39	**57.73** ± 0.21
Speedup	1.0	3.21	1.16	2.98	1.19	3.18	6.1

The best metric values are in bold

Table 3 Classification results obtained for the SV dataset, using 10% randomly selected samples to perform the training stage

Class	Vanilla RNN		LSTM		GRU		Proposed
	CUDA	CuDNN	CUDA	CuDNN	CUDA	CuDNN	
1	99.63 ± 0.09	99.69 ± 0.42	99.83 ± 0.09	99.63 ± 0.45	99.91 ± 0.09	**99.97** ± 0.07	99.7 ± 0.23
2	99.97 ± 0.03	99.88 ± 0.18	99.83 ± 0.15	99.98 ± 0.03	99.97 ± 0.03	**99.98** ± 0.01	99.96 ± 0.03
3	97.46 ± 1.49	98.19 ± 0.45	**99.73** ± 0.14	99.46 ± 0.33	98.66 ± 1.21	99.56 ± 0.37	98.82 ± 0.58
4	98.69 ± 0.62	99.17 ± 0.28	**99.44** ± 0.24	99.3 ± 0.24	98.87 ± 0.35	99.23 ± 0.26	99.31 ± 0.44
5	98.38 ± 0.53	98.24 ± 0.57	99.05 ± 0.41	99.15 ± 0.49	99.04 ± 0.53	**99.27** ± 0.34	98.74 ± 0.48
6	99.87 ± 0.1	99.81 ± 0.12	**99.9** ± 0.07	99.86 ± 0.09	99.88 ± 0.08	99.89 ± 0.1	99.89 ± 0.09
7	99.78 ± 0.14	99.69 ± 0.17	99.75 ± 0.1	**99.84** ± 0.2	99.79 ± 0.1	99.7 ± 0.11	99.76 ± 0.11
8	88.36 ± 1.04	89.42 ± 2.27	88.7 ± 1.98	**89.8** ± 0.97	88.9 ± 1.76	89.34 ± 0.6	88.33 ± 1.6
9	99.81 ± 0.14	99.8 ± 0.11	99.72 ± 0.2	99.83 ± 0.11	99.91 ± 0.1	**99.98** ± 0.01	99.86 ± 0.08
10	96.01 ± 0.43	96.17 ± 0.92	97.57 ± 0.32	97.57 ± 0.51	97.77 ± 0.49	**97.9** ± 0.23	96.61 ± 0.77
11	98.09 ± 0.75	96.75 ± 0.38	97.77 ± 1.84	98.06 ± 1.13	**99.33** ± 0.25	97.86 ± 1.08	98.38 ± 0.47
12	99.53 ± 0.31	99.08 ± 0.91	**99.88** ± 0.1	99.7 ± 0.47	99.64 ± 0.21	99.86 ± 0.11	99.71 ± 0.2
13	98.71 ± 0.76	98.54 ± 0.43	**98.88** ± 0.75	98.42 ± 0.92	98.86 ± 0.69	98.81 ± 0.49	98.76 ± 0.44
14	96.74 ± 1.14	97.09 ± 1.58	97.11 ± 1.77	97.63 ± 1.06	97.45 ± 0.71	**98.09** ± 0.89	96.91 ± 1.14
15	73.31 ± 1.12	69.99 ± 5.36	76.94 ± 3.22	75.72 ± 0.62	**77.62** ± 2.16	77.32 ± 0.99	69.94 ± 2.63
16	99.25 ± 0.32	**99.32** ± 0.28	99.07 ± 0.35	99.09 ± 0.38	99.07 ± 0.57	99.19 ± 0.61	99.03 ± 0.3
OA	93.32 ± 0.21	93.08 ± 0.31	94.1 ± 0.22	94.18 ± 0.18	94.26 ± 0.25	**94.37** ± 0.06	93.0 ± 0.09
AA	96.47 ± 0.11	96.3 ± 0.28	97.07 ± 0.24	97.07 ± 0.05	97.17 ± 0.12	**97.25** ± 0.12	96.48 ± 0.18
K(x100)	92.55 ± 0.23	92.29 ± 0.35	93.43 ± 0.24	93.51 ± 0.2	93.6 ± 0.28	**93.73** ± 0.07	92.19 ± 0.1
Parameters	239312		252176		247888		**235536**
Runtime(s)	564.77 ± 1.9	199.83 ± 2.09	552.8 ± 1.88	216.65 ± 1.54	567.29 ± 1.93	198.25 ± 1.51	**74.13** ± 0.31
Speedup	1.0	2.84	1.03	2.62	1.0	2.86	7.65

The best metric values are in bold

Focusing on the number of parameters and run times, we observe that both the CUDA and CuDNN versions require the same number of parameters. However, their run times are significantly different, being CuDNN faster. With this in mind, our model exhibits the lowest number parameters, resulting in less computational requirements, lower memory consumption and fast performance. In fact, the SRU is able to outperform all the optimized models in CuDNN. In order to analyze the speedup achieved by these methods, we first note that the slowest one is the most basic model (i.e., the vanilla RNN directly implemented in CUDA). Further, the CUDA versions of LSTM and GRU achieve low speedups, while the CuDNN counterparts achieve an approximate speedup of x3 for the IP, BIP and UP dataset, and x2 for the SV scene. However, the speedups achieved by the proposed method are close to x5, x6 and x7 for the IP, UP and SV scenes, respectively, and more than x10 for the BIP. This shows the improved computational performance (and scalability with size) of the proposed SRU implementation.

4.4.2 Experiment 2: speedup and scalability

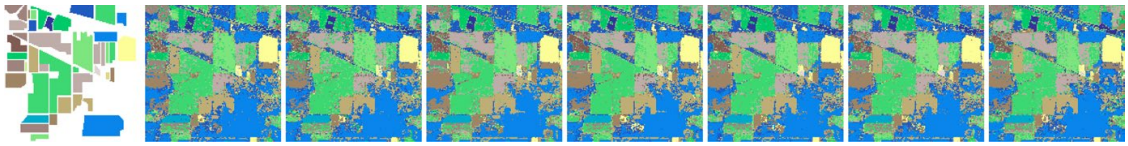
The results of our second experiment are reported on the last rows of Tables 1, 2 and 3. Once more, the CUDA implementation of vanilla RNN was the slowest model, and we used it as a baseline to evaluate the other implementations. If we focus on the CUDA LSTM and GRU, their speedup never exceeds x1.20 and even decreases when more input features are used (e.g., for the IP and SV scenes). In turn, the CuDNN-based models exhibit speedup values of x2 and x3, being the CuDNN-based vanilla model the one with the highest speedup (closely followed by the GRU network). However, these implementations do not scale well when additional input features are considered, even reducing their speedup in some cases (e.g., for the SV scene). In this sense, the proposed method not only reaches the highest speedup values, but the associated speedups always increase with the number of input features (for instance in BIP) (Table 4).

Figure 6 shows a graphical representation of the runtimes measured (in s) for the different tested methods, as a function of the number of input features (spectral bands). As we can observe, the CUDA versions of vanilla, LSTM and GRU are clearly the slowest methods for the considered HSI datasets. These models are all negatively affected by

Table 4 Classification results obtained for the BIP dataset, using 15% randomly selected samples. Due to space limitations, the accuracy of each class is not shown

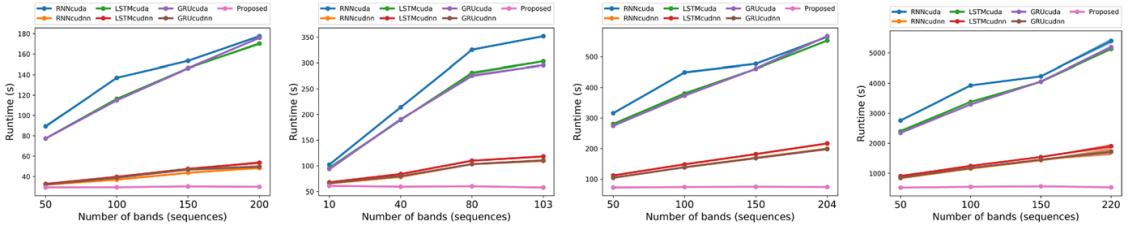
Class	Vanilla RNN		LSTM		GRU		Proposed
	CUDA	CuDNN	CUDA	CuDNN	CUDA	CuDNN	
OA	57.9 ± 0.49	58.12 ± 0.33	60.71 ± 1.39	60.28 ± 0.51	**61.6** ± 0.33	61.38 ± 0.64	56.73 ± 0.22
AA	46.4 ± 0.58	46.09 ± 0.43	49.16 ± 1.14	48.31 ± 1.1	**50.83** ± 0.68	49.47 ± 0.93	43.42 ± 0.55
K(x100)	54.34 ± 0.53	54.57 ± 0.34	57.42 ± 1.51	56.92 ± 0.54	**58.42** ± 0.4	58.1 ± 0.69	53.04 ± 0.21
Parameters	849146		862010		857722		**845370**
Runtime(s)	5327.74 ± 107.52	1759.85 ± 107.58	5076.23 ± 77.47	1934.94 ± 68.97	5180.01 ± 52.87	1745.2 ± 89.31	**508.19** ± 56.77
Speedup	1.0	3.03	1.05	2.75	1.03	3.05	**10.48**

The best metric values are in bold



(a) Ground-truth **(b)** V-RNN CUDA **(c)** V-RNN CuDNN **(d)** LSTM CUDA **(e)** LSTM CuDNN **(f)** GRU CUDA **(g)** GRU CuDNN **(h)** Proposed

Fig. 5 Classification maps of IP scene, using 15% randomly selected samples to perform the training stage



(a) Indian Pines (IP) **(b)** University of Pavia (UP) **(c)** Salinas Valley (SV) **(d)** Big Indian Pines (BIP)

Fig. 6 Graphical representation of the runtimes (in s) measured for the different tested methods, as a function of the number of input features (spectral bands)

the increase in the number of input features (i.e., the runtimes increase significantly with the number of input features). However, the CuDNN counterparts exhibit better computational performance and scalability than CUDA versions when the number of input features increases. Finally, the proposed SRU-based model clearly exhibits the lowest runtimes (which do not increase with the number of features), thanks to the optimal parallelization of its point-wise operations. As a result, our method is not affected by the inclusion of additional input features, scaling significantly better than the other tested methods.

5 Conclusions and future work

In this paper, a new RNN model for HSI data classification is presented and discussed. The proposed model is based on the SRU architecture, which reduces the internal complexity of other previously developed recurrent approaches (i.e., LSTM and GRU) by decoupling the computational relationship between the current and previous states in the network architecture. This is achieved by resorting to easily parallelizable, point-wise operations. Our experiments, conducted using four benchmark HSI datasets, reveal that our method is able to achieve good classification results using much fewer parameters than the traditional models (vanilla, LSTM and GRU), therefore consuming less memory. Moreover, our parallelization strategy significantly reduces the measured runtimes of the proposed method, which obtains higher speedups as the number of pixels (and the dimensionality of input features) in the HSI increase.

As future work, we will study the inclusion of standardization and regularization methods to reduce the overfitting of the proposed model, with the goal of further improving the reliability and precision of the obtained classification results.

Acknowledgements This study was supported by Spanish Ministry (FPU14/02012-FPU15/02090, ESP2016-79503-C2-2-P), FEDER and Junta de Extremadura (GR18060), and the European Union (734541-EXPOSURE).

References

1. Appleyard J, Kocisky T, Blunsom P (2016) Optimizing performance of recurrent neural networks on gpus. arXiv:1604.01946
2. Campos V, Jou B, Giró-i Nieto X, Torres J, Chang SF (2017) Skip rnn: learning to skip state updates in recurrent neural networks. arXiv:1708.06834
3. Chen Y, Lin Z, Zhao X, Wang G, Gu Y (2014) Deep learning-based classification of hyperspectral data. IEEE J Select Top Appl Earth Obs Remote Sens 7(6):2094–2107
4. Cho K, Van Merriënboer B, Bahdanau D, Bengio Y (2014) On the properties of neural machine translation: encoder-decoder approaches. arXiv:1409.1259
5. Cudahy T, Hewson R, Huntington J, Quigley M, Barry P (2001) The performance of the satellite-borne hyperion hyperspectral VNIR-SWIR imaging system for mineral mapping at Mount Fitton, South Australia. In: IGARSS 2001. Scanning the present and resolving the future. Proceedings. IEEE 2001 international geoscience and remote sensing symposium (Cat. No. 01CH37217). IEEE, vol 1, pp 314–316
6. Cybenko G (1989) Approximation by superpositions of a sigmoidal function. Math Control Signals Syst 2(4):303–314
7. Gevaert CM, Suomalainen J, Tang J, Kooistra L (2015) Generation of spectral-temporal response surfaces by combining multispectral satellite and hyperspectral UAV imagery for precision agriculture applications. IEEE J Select Top Appl Earth Obs Remote Sens 8(6):3140–3146
8. Goetz AF, Vane G, Solomon JE, Rock BN (1985) Imaging spectrometry for earth remote sensing. Science 228(4704):1147–1153
9. Green RO, Eastwood ML, Sarture CM, Chrien TG, Aronsson M, Chippendale BJ, Faust JA, Pavri BE, Chovit CJ, Solis M, Olah MR, Williams O (1998) Imaging spectroscopy and the airborne visible/infrared imaging spectrometer (AVIRIS). Remote Sens Environ 65(3):227–248
10. Hochreiter S (1998) Recurrent neural net learning and vanishing gradient. Int J Uncertain Fuzziness Knowl-Based Syst 6(2):107–116
11. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
12. Kumar DN, Reshmidevi T (2013) Remote sensing applications in water resources. J Indian Inst Sci 93(2):163–188
13. Kunkel B, Blechinger F, Lutz R, Doerffer R, van der Piepen H, Schroder M (1988) ROSIS (Reflective Optics System Imaging Spectrometer)—A candidate instrument for polar platform missions. In: Seeley J, Bowyer S (eds) Proceedings of SPIE 0868 optoelectronic technologies for remote sensing from space, p 8 . https://doi.org/10.1117/12.943611
14. Landgrebe D (2002) Hyperspectral image data analysis. IEEE Signal Process Mag 19(1):17–28
15. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436
16. Lei T, Zhang Y, Wang SI, Dai H, Artzi Y (2018) Simple recurrent units for highly parallelizable recurrence. In: Proceedings of the 2018 conference on empirical methods in natural language processing, Brussels, Belgium, October 31–November 4, 2018, pp 4470–4481. https://doi.org/10.18653/v1/d18-1477
17. Li B, Zhou E, Huang B, Duan J, Wang Y, Xu N, Zhang J, Yang H (2014) Large scale recurrent neural network on gpu. In: 2014 International Joint Conference on Neural Networks (IJCNN). IEEE, pp 4062–4069
18. Li Z, Chen J, Baltsavias E (2008) Advances in photogrammetry, remote sensing and spatial information sciences: 2008 ISPRS congress book, vol 7. CRC Press, Boca Raton
19. Manolakis D, Shaw G (2002) Detection algorithms for hyperspectral imaging applications. IEEE Signal Process Mag 19(1):29–43
20. Meerdink SK, Roberts DA, Roth KL, King JY, Gader PD, Koltunov A (2019) Classifying California plant species temporally using airborne hyperspectral imagery. Remote Sens Environ 232:111308

21. Mou L, Ghamisi P, Zhu XX (2017) Deep recurrent neural networks for hyperspectral image classification. IEEE Trans Geosci Remote Sens 55(7):3639–3655
22. Murphy RJ, Monteiro ST, Schneider S (2012) Evaluating classification techniques for mapping vertical geology using field-based hyperspectral sensors. IEEE Trans Geosci Remote Sens 50(8):3066–3080
23. Nurvitadhi E, Sim J, Sheffield D, Mishra A, Krishnan S, Marr D (2016) Accelerating recurrent neural networks in analytics servers: comparison of FPGA, CPU, GPU, and ASIC. In: 2016 26th International Conference on Field Programmable Logic and Applications (FPL). IEEE, pp 1–4
24. Paoletti ME, Haut JM, Plaza J, Plaza A (2019) Deep learning classifiers for hyperspectral imaging: a review. ISPRS J Photogramm Remote Sens 158:279–317. https://doi.org/10.1016/j.isprsjprs.2019.09.006
25. Plaza A, Du Q, Chang YL, King RL (2011) High performance computing for hyperspectral remote sensing. IEEE J Select Top Appl Earth Obs Remote Sens 4(3):528–544
26. Srivastava RK, Greff K, Schmidhuber J (2015) Training very deep networks. In: Advances in neural information processing systems, pp 2377–2385
27. Weber C, Aguejdad R, Briottet X, Avala J, Fabre S, Demuynck J, Zenou E, Deville Y, Karoui MS, Benhalouche FZ et al (2018) Hyperspectral imagery for environmental urban planning. In: IGARSS 2018-2018 IEEE international geoscience and remote sensing symposium. IEEE, pp 1628–1631
28. Weninger F, Bergmann J, Schuller B (2015) Introducing currennt: the Munich open-source CUDA recurrent neural network toolkit. J Mach Learn Res 16(1):547–551
29. Williams RJ, Zipser D (1989) A learning algorithm for continually running fully recurrent neural networks. Neural Comput 1(2):270–280
30. Wold S, Esbensen K, Geladi P (1987) Principal component analysis. Chemom Intell Lab Syst 2(1–3):37–52
31. Xu Y, Zhang L, Du B, Zhang F (2018) Spectral–spatial unified networks for hyperspectral image classification. IEEE Trans Geosci Remote Sens 56(10):5893–5909
32. Yang X, Ye Y, Li X, Lau RY, Zhang X, Huang X (2018) Hyperspectral image classification with deep learning models. IEEE Trans Geosci Remote Sens 56(9):5408–5423
33. Zhang X, Sun Y, Jiang K, Li C, Jiao L, Zhou H (2018) Spatial sequential recurrent neural network for hyperspectral image classification. IEEE J Select Top Appl Earth Obs Remote Sens 11(11):4141–4155
34. Zhou F, Hang R, Liu Q, Yuan X (2019) Hyperspectral image classification using spectral-spatial LSTMs. Neurocomputing 328:39–47

Escuela Politecnica
Av. de la Universidad, S/N, 10003
Caceres, Spain
Phone: 0034927257000. Ext. 51662
Email: aplaza,jplaza{@unex.es}

Dr. Antonio Plaza Miguel y Dr. Javier Plaza Miguel como directores de la tesis titulada "Procesamiento eficiente y profundo de imgenes hiperespectrales de la observación remota de la Tierra y aplicaciones en tareas de clasificación", certifico el factor de impacto y la categorización de la siguiente publicación, incluida en la tesis doctoral. Del mismo modo, se especifica la aportación del doctorado. Si necesitas cualquier información o clarificación, por favor, no dude en contactar conmigo.

Antonio Plaza Miguel PhD. and Javier Plaza Miguel PhD as directors of the Phd thesis titled "Deep-efficient processing of remote sensing hyperspectral images and applications in classification tasks.", certify the impact factor and the categorization of the following publication, included in the doctoral thesis. In the same way, the contribution of the doctorate is specified. If you need any further information or clarification, please do not hesitate contacting me.

Artculo / Paper

Autores/Authors: **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza.
Title: A New Deep Convolutional Neural Network for Fast Hyperspectral Image Classification.
Journal: ISPRS Journal of Photogrammetry and Remote Sensing.
Other Information: vol. 145, Part A, pp. 120-147, November 2018.
DOI: 10.1016/j.isprsjprs.2017.11.021.
Impact factor 2018: 6.942. Q1
Abstract: Artificial neural networks (ANNs) have been widely used for the analysis of remotely sensed imagery. In particular, convolutional neural networks (CNNs) are gaining more and more attention in this field. CNNs have proved to be very effective in areas such as image recognition and classification, especially for the classification of large sets composed by two-dimensional images. However, their application to multispectral and hyperspectral images faces some challenges, especially related to the processing of the high-dimensional information contained in multidimensional data cubes. This results in a significant increase in computation time. In this paper, we present a new CNN architecture for the classification of hyperspectral images. The proposed CNN is a 3-D network that uses both spectral and spatial information. It also implements a border mirroring strategy to effectively process border areas in the image, and has been efficiently implemented using graphics processing units (GPUs). Our experimental results indicate that the proposed network performs accurately and efficiently, achieving a reduction of the computation time and increasing the accuracy in the classification of hyperspectral images when compared to other traditional ANN techniques.

Contribución del doctorado: Planteamiento de la hipótesis, desarrollo práctico, análisis y discusión de los resultados, elaboración y escritura del manuscrito.

Firma / Signature
Mayo / May, 2020

Antonio Plaza Miguel

Javier Plaza Miguel

ISPRS Journal of Photogrammetry and Remote Sensing 145 (2018) 120–147

Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

journal homepage: www.elsevier.com/locate/isprsjprs

A new deep convolutional neural network for fast hyperspectral image classification

M.E. Paoletti *, J.M. Haut, J. Plaza, A. Plaza

Hyperspectral Computing Laboratory (HyperComp), Department of Computer Technology and Communications, Escuela Politecnica de Caceres, University of Extremadura, Avenida de la Universidad s/n, E-10002 Caceres, Spain

ARTICLE INFO

Article history:
Received 31 May 2017
Received in revised form 27 October 2017
Accepted 29 November 2017
Available online 6 December 2017

Keywords:
Hyperspectral imaging
Deep learning
Convolutional neural networks (CNNs)
Classification
Graphics processing units (GPUs)

ABSTRACT

Artificial neural networks (ANNs) have been widely used for the analysis of remotely sensed imagery. In particular, convolutional neural networks (CNNs) are gaining more and more attention in this field. CNNs have proved to be very effective in areas such as image recognition and classification, especially for the classification of large sets composed by two-dimensional images. However, their application to multispectral and hyperspectral images faces some challenges, especially related to the processing of the high-dimensional information contained in multidimensional data cubes. This results in a significant increase in computation time. In this paper, we present a new CNN architecture for the classification of hyperspectral images. The proposed CNN is a 3-D network that uses both spectral and spatial information. It also implements a border mirroring strategy to effectively process border areas in the image, and has been efficiently implemented using graphics processing units (GPUs). Our experimental results indicate that the proposed network performs accurately and efficiently, achieving a reduction of the computation time and increasing the accuracy in the classification of hyperspectral images when compared to other traditional ANN techniques.

1. Introduction

Remote sensing image acquisition and processing has become very important in recent times in Earth observation problems, exhibiting many practical applications such as monitoring and management of the environment, agriculture or security and defense/intelligence issues. Of particular importance is the computationally efficient processing of images formed by multiple spectral bands, called multispectral and hyperspectral images. These kinds of images collect information corresponding to large observation areas on the surface of the Earth, using dozens/hundreds of contiguous spectral bands (Chang, 2003), thus creating a three-dimensional data cube with size significantly larger than traditional remotely sensed images. As a result, multispectral and hyperspectral images require particular computational improvements, especially for their storage and advanced processing.

Several methods have been developed for fast processing and classification of multispectral and hyperspectral images (Cheng et al., 2017; Yuan et al., 2015), from those that only use spatial or spectral information to those that combine both kinds of data. This includes unsupervised techniques such as clustering (Haut et al., 2017a; Tarabalka et al., 2009; Paoletti et al., 2017). However, supervised classifiers are often preferred, due to their capacity to provide high classification accuracies, although these methods may be affected by the limited availability of training samples as they generally need a large number of samples in order to obtain those good results. In particular, supervised methods face challenges in the classification of hyperspectral images due to the unbalance between the high dimensionality of the data and the limited number of training samples available in practice (*Hughes phenomenon*) (Khodadadzadeh et al., 2014). In this sense, support vector machines (SVMs) (Scholkopf and Smola, 2001) and multinomial logistic regression (MLR) (Böhning, 1992) have been proved to be very useful for the supervised classification of hyperspectral images due to their ability to deal with large input spaces (Melgani and Bruzzone, 2004; Fauvel et al., 2008; Plaza et al., 2009; Camps-Valls and Bruzzone, 2005; Wu et al., 2015; Haut et al., 2017b). Also, some sampling query strategies have been proposed to address the limited availability of training samples, such as semi-supervised and active learning methods (Li et al., 2010, 2011; Rajan et al., 2008).

At this point, we can highlight several spatial-spectral classification methods that combine the strengths of semi-supervised methods and active learning techniques, for example those based on morphological component analysis (MCA) (Starck et al., 2005), a method that decomposes images into texture and cartoon

* Corresponding author.
E-mail address: mpaolett@alumnos.unex.es (M.E. Paoletti).

https://doi.org/10.1016/j.isprsjprs.2017.11.021

(piecewise smooth) parts. In Zhou and Prasad (2017), authors presented a new framework that combines active and semi-supervised learning with MCA for hyperspectral image classification. Also, in Xu et al. (2016b) authors presented a new classification framework for the fusion of hyperspectral and light detection and ranging (LiDAR) data, combining MCA for textural feature extraction and MLR for classification purposes. The multiple MCA (MMCA) (Xu et al., 2016a) is an extension of the MCA that uses both spatial and spectral features. Its goal is to separate an image into two components: a smoothness component and a texture component.

On the other hand, due to their success in the field of pattern recognition (Bishop, 1995; Atkinson and Tatnall, 1997) and the availability of multiple training techniques (including machine learning, deep learning and active learning techniques, as well as supervised, unsupervised and semi-supervised approaches) to deal with linearly non-separable data (Benediktsson and Swain, 1990), artificial neural networks (ANNs) have attracted the attention of a large number of researchers in the area of hyperspectral image classification and analysis (Benediktsson et al., 1993; Yang, 1999) as compared to probabilistic methods. In particular, we highlight the use of convolutional neural networks (CNNs) (LeCun et al., 1998a) as a powerful deep learning model for image classification, which can effectively combine the spatial and spectral information.

1.1. Deep learning and CNNs: a review

For years, building a machine learning system required a great deal of effort in designing a feature extractor that would transform raw data (i.e. pixel values from an image) into a feature vector from which the learning subsystem could detect/classify patterns (LeCun et al., 2015). Deep learning (or deep structured learning) emerged in 2006 with deep belief networks (DBNs)[1] (Hinton et al., 2006; Hinton and Salakhutdinov, 2006) as a part of a machine learning system that exploits many layers of non-linear information processing for supervised or unsupervised feature extraction and transformation, and also for pattern analysis and classification (Bengio, 2009).

After DBNs, two new unsupervised deep models were developed: (1) a method for learning sparse, overcomplete features that uses a linear encoder-decoder preceded by a sparsifying non-linearity that turns a code vector into a quasi-binary sparse code vector (Ranzato et al., 2006) and (2) a variant of autoencoder with greedy layer-wise training (Bengio et al., 2007).

With the advancement of technology (both hardware and software) and the development of new optimization algorithms[2] (LeCun et al., 1998b) new milestones were achieved in deep learning, giving as result three types of deep methods:

- Unsupervised deep networks (generative learning): these methods work without labeled classes, looking for patterns between pixels through capturing high-order correlation of data (e.g. autoencoder-based methods[3] (Licciardi and Del Frate, 2011; Chen et al., 2014), RBMs (Midhun et al., 2014), DBNs (Li et al., 2014b), and deep Boltzmann machines or DBMs (Salakhutdinov and Hinton, 2009; Wu et al., 2016)).
- Supervised deep networks (discriminative deep networks): these models work with labeled information and their goal is to categorize the input data in these labels. They represent the most common form of machine learning, deep or not (LeCun et al., 2015), and these kinds of models are usually more efficient to train and test, more flexible to construct, and more suitable for end-to-end learning of complex systems (Deng and Yu, 2014). We can distinguish between linear supervised deep method[4] (e.g. deep neural networks or DNNs[5] with linear activation functions) and non-linear supervised methods (e.g. deep stacking networks or DSNs (He et al., 2016), recurrent neural networks or RNNs[6] and Convolutional neural networks or CNNs (LeCun et al., 2015)).[7]
- Hybrid deep networks (semisupervised methods): these methods make use of both generative and discriminative model components, i.e., they work with and without labeled data (e.g. generative adversarial networks or GANs (Goodfellow et al., 2014)). Semi-supervised learning is very useful in hyperspectral image classification in order to deal with the limited training samples problem (Ma et al., 2016).

Focusing on CNNs, these supervised non-linear models are a special type of deep learning model that is inspired by neuroscience (Ghamisi et al., 2017) and are designed to process data that come in the form of multiple arrays. The literature on CNN applied to remote sensing classification shows different points of view in the way these models are used. Basically there are three ways to apply CNNs:

1. Extracting only spectral information: spectral-based classification approaches are conceptually simple and easy to be implemented, but they neglect the spatial components (Li et al., 2014a). Normally, these methods assume that each pixel is pure and typically labeled as a single land use and land cover type (Fisher, 1997). For spectral feature classification with CNNs, the spectral feature of the original image data is directly deployed as the input vector (Zhang et al., 2016a), so we obtain a 1-D CNN architecture that receives $N \times 1$ input vectors, where N is the number of spectral bands (Ghamisi et al., 2017; Hu et al., 2015b; Chen et al., 2016).
2. Extracting only spatial information: these models consider the neighboring pixels of a certain pixel in the original remote sensing image in order to extract the spatial feature representation (Zhang et al., 2016a). As a result, 2-D CNN architectures are adopted, where the input data is a patch of $P \times P$ neighboring pixels (Vetrivel et al., 2018; Chen et al., 2016; Hu et al., 2015a). In this sense, several methods have been implemented in order to extract high-level spatial features, as multi-scale image information (Liu et al., 2016; Zhao and Du, 2016a; Zhang et al., 2016b; Yu et al., 2017). For hyperspectral image

[1] A DBN is composed by a stack of restricted Boltzmann machines (Smolensky, 1986; Larochelle and Bengio, 2008) (RBMs). The DBN core is a greedy learning algorithm that optimizes the network weights layer by layer. Its complexity grows linearly with the size and depth of the network.

[2] For example, new variants of gradient descent optimizer were developed, including batch and mini-batch gradient descent, the stochastic gradient descent (SGD) or the included momentum in SGD (Qian, 1999). New optimizers also appear as Adagrad optimizer (Duchi et al., 2011) and its extension Adadelta (Zeiler, 2012) or the Adam optimizer (Kingma and Ba, 2014).

[3] An autoencoder (Cho, 2014; Karhunen et al., 2015) is a neural network (or mapping method) where the desired output is equal to the input data vector. We can distinguish between linear autoencoders, with only one hidden layer that works like a principal component analysis (PCA) if its weights between the encoder and decoder are tied, and non-linear or deep autoencoders, which have more modeling power by employing multiple nonlinear intermediate layers symmetrically in the encoder and decoder (Cho, 2014).

[4] Linear classifiers have an important limitation: these methods can only carve their input space into very simple regions (half-spaces) separated by a hyperplane (Chien, 1974).

[5] A DNN is a multi-layer perceptron (MLP) with many hidden and fully connected layers.

[6] RNNs can also be used as generative learning models if the output is not a label sequence associated with the input data sequence. Long short-term memory networks (LSTMs) are a kind of RNN (Hochreiter and Schmidhuber, 1997).

[7] CNNs can also work in unsupervised and semi-supervised mode (Dosovitskiy et al., 2014; Romero et al., 2016; Liu et al., 2017). On the other hand, recent efforts have resulted in deconvolutional neural networks (DCNN) (Zeiler et al., 2010). In Lu et al. (2017) authors combined the spatial pyramid model (SPM) with a shallow weighted DCNN to learn a set of feature maps and filters by minimizing the reconstruction error between the input image and the convolution result.

analysis normally it is necessary a pre-processing of the spectral information, with reduction of the number of bands for example (using, e.g., PCAs or autoencoders).

3. Extracting spectral-spatial information: the use of spatial features with spectral information in combined fashion can significantly improve the classification accuracy. When we talk about extracting spectral-spatial information with CNNs, two types of models can be highlighted:
   - Those models that mix various techniques in addition to CNNs to extract spectral-spatial information separately and then combine them, for example using a 1-D CNN and 2-D CNN (Zhang et al., 2017; Yang et al., 2016) or combining different spectral feature extractors with spatial CNNs (Zhao and Du, 2016b). These methods do not take full advantage of the joint spatial/spectral correlation information.
   - 3-D CNN architectures (Chen et al., 2016; Li et al., 2017) that compute each pixel in association with a $P \times P$ spatial neighborhood and B spectral bands ($P \times P \times B$). These models apply 3-D kernels in order to learn the local signal changes in both the spatial and the spectral domain of the hyperspectral data cubes, exploiting important discriminative information for classification and taking full advantage of the structural characteristics of the 3-D remote sensing data in general and hyperspectral images in particular (Li et al., 2017).

In this work, we propose an improved 3-D deep CNN model composed by 5 layers which uses all the spatial-spectral information of the hyperspectral image. We also include a specific strategy for management of the borders of the image and further develop an efficient implementation in graphics processing units (GPUs) to significantly speed up the computational performance. Deep Learning techniques, and in particular CNNs, involve a huge amount of matrix and vector operations. Most of these operations can be easily and massively parallelized using GPUs, due to their inherent design with hundreds/thousands of cores that can compute one or several matrix operations in parallel which, compared to a CPU with a few cores, results in a important decrease in computation time.

As a result, the main contributions of our work can be highlighted as follows: (1) the development of a new CNN architecture that considers the spatial and spectral information contained in hyperspectral images in simultaneous fashion, and (2) the development of an efficient implementation of the newly proposed architecture on GPUs that allows for efficient exploitation of the proposed methodology in real applications.

The remainder of the paper is organized as follows. Section 2 provides some general aspects about CNNs. Section 3 describes the proposed CNN. Section 4 validates the proposed approach by comparing it with classic ANNs such as the MLP and other CNN implementations in the literature, in order to illustrate the advantages of the proposed implementation in terms of both computational efficiency and classification accuracy. Section 5 concludes the paper with some remarks and hints at plausible future research lines.

2. CNNs

CNNs are composed by a set of blocks that can be applied both across space and across time (images, audio and video signals). Each block transforms the input volume to an output volume of neuron activations which will serve as input to the next block. In contrast to conventional ANNs, the blocks of neurons in CNNs operate like kernels which are connected and applied over one patch of the input volume (see Fig. 1), that is, the neurons of a block

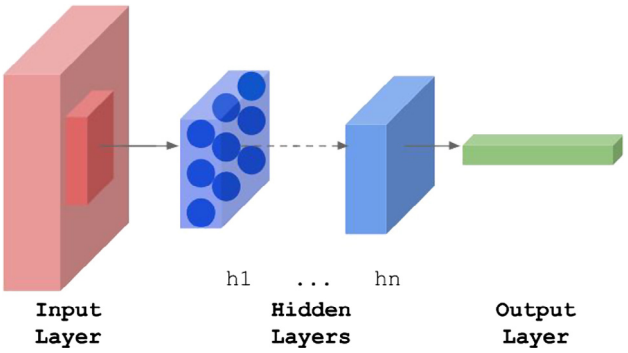


Fig. 1. CNN architecture. Each block or layer of a CNN transforms the input volume to an output volume of neuron activations. Neurons in layer l are connected to a small region of layer $l-1$.

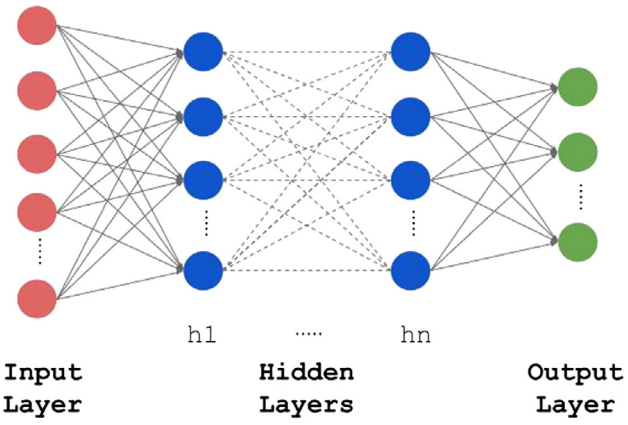


Fig. 2. MLP architecture. All nodes in one layer are fully connected with the nodes of the previous layer.

are not fully-connected to all neurons of the previous layer as in the standard MLP (see Fig. 2). These blocks actually compose feature extraction stages, which specifically consist of three layers (Zhang et al., 2016a) that are the key parts of almost all CNN models:

1. Convolution layer: a 3-D layer where each neuron computes the dot product between its weights and a small region of the input volume, i.e. a rectangular section of the previous layer, to which it is connected. Its goal is to identify certain features from the previous layer and mapping their appearance to a feature map (LeCun et al., 2015). We can see this layer as set of k filters of size $l \times l \times q$ (filter bank) where the neurons share the same weights and bias and connect the input volume to the output volume (Zhang et al., 2016a). Each filter detects a particular feature at every location on the input. The resulting output volume of the layer l is a feature map of size $d^l \times d^l \times k^l$ that stores the information where the feature occurs in the original input volume and is calculated as $z_i^l = B^l + \sum_{j=1}^{k^{l-1}} W_{i,j}^l * z_j^{l-1}$, where $i \in [1, k^l]$, B^l is the bias matrix of layer l and $W_{i,j}^l$ is the weight matrix or filter (also known as kernel or feature detector)[8] that connects the jth feature map in layer $l-1$ (z_j^{l-1}) with the ith feature map in layer l.

[8] The concept of weight matrix can be understood as a feature detector or filter which can beused to search for an specific spatial characteristic of the input data. The weight matrix will assign a greater weight to the pixels that collect this characteristic penalizing the pixels that do not exhibit this spatial behaviour.

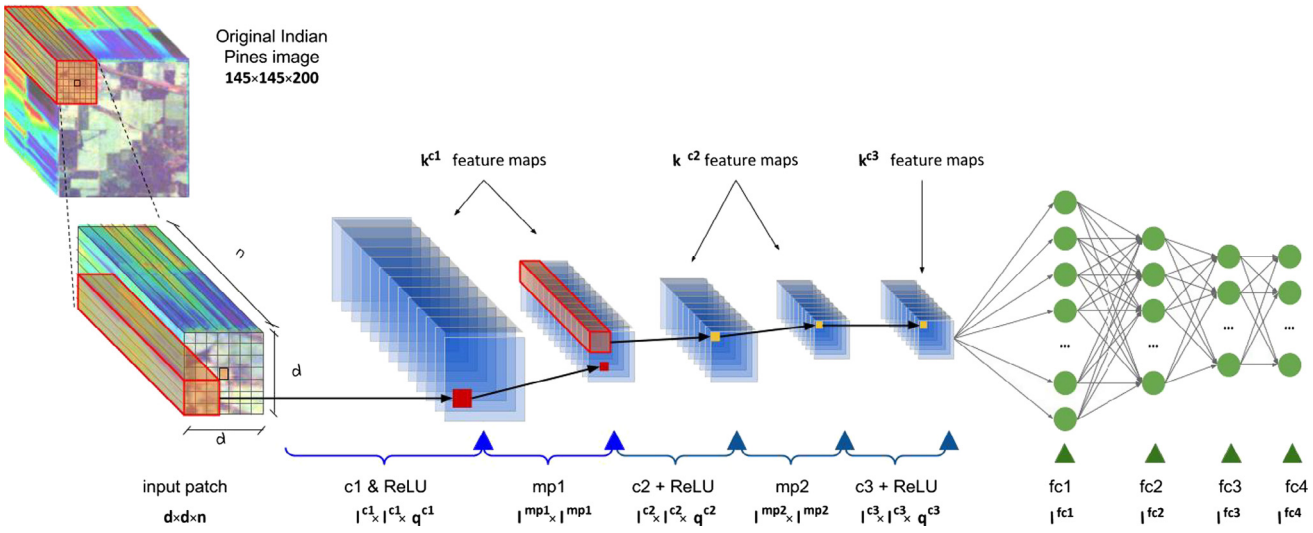


Fig. 3. Proposed CNN architecture.

2. Nonlinearity layer: this layer embeds a nonlinear function (as the rectified linear unit or ReLU). Jarrett et al. (2009), Nair and Hinton (2010), and Glorot et al. (2011) that is applied to each feature map's component in order to learn nonlinear representations: $a^l = f(z^l)$.
3. Pooling layer: this layer is used to make the features invariant from the location and to summarize the output of multiple neurons in convolution layers through a pooling function. In our case, this layer executes a max operation within a small spatial region R over the resulting feature map after the nonlinearity layer: $p^l = \max_{i \in R} a_i^l$.

Sparse connectivity and shared weights make CNNs ideal for processing and classifying images, reducing the number of parameters to be learned by the network and ensuring some degree of shift, scale, and distortion in-variance.

3. Proposed CNN

The structure of our new CNN is shown in Fig. 3. As we can see, our CNN consists of an input layer, three convolution layers with ReLU as nonlinear activation function, two maxpool layers, and four fully-connected layers. The last one is the output layer which obtains the desired label for the input data. In the following, we first provide details about our preprocessing strategy for hyperspectral data, particularly to account for spatial information, prior to feeding this information to the CNN. Then, we provide a detailed explanation about the considered architecture.

3.1. Data preprocessing

Normally, CNNs receive a complete normalized image prior to classification. However, in hyperspectral images the classes are

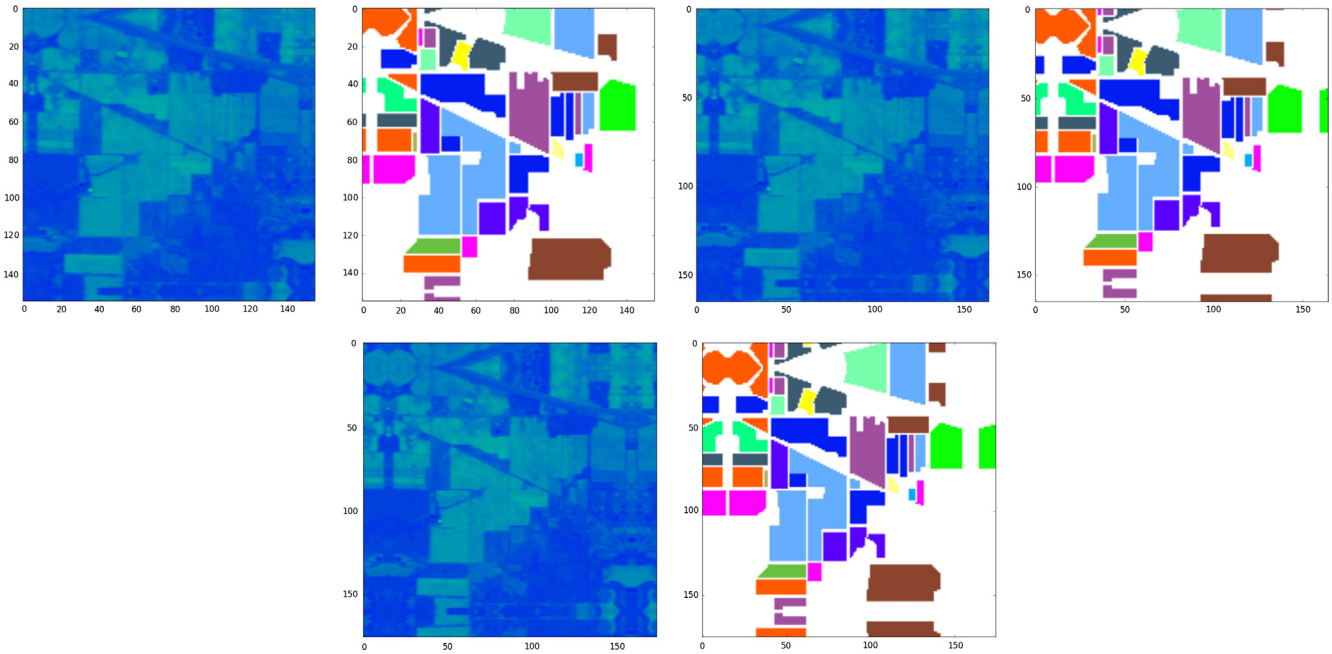


Fig. 4. Graphical illustration of our border mirroring strategy (with $d = 9, d = 19$ and $d = 29$) using the well-known AVIRIS Indian Pines hyperspectral image (and its associated ground-truth) as an example.

typically mixed within the image, so we feed the pixel (vectors) one by one to the network. This allows exploiting the rich spectral information contained in the hyperspectral data, but we need an additional mechanism in order to include also the spatial information.

To achieve this and take advantage of both the spatial and the spectral information simultaneously, we have implemented a 3-D approach in which we feed the network with a neighborhood window centered around each pixel vector. In this way, the input layer accepts volumes of size $d \times d \times n$, where d is the width and height of the input volume and n is the total number of bands of the original hyperspectral image. This requires a pre-processing stage to divide the hyperspectral image into patches of size $d \times d \times n$. The desired label to be reached by the network will be the one that owns the central pixel of the patch $[d/2+1, d/2+1, n]$.

However, this preprocessing strategy faces a problem: for the pixels belonging to the borders of the image a $d \times d$ surrounding neighborhood cannot be defined. As we increase d, we cannot properly account for the border information around some pixels. Some approaches simply disregard those border pixels for which they do not have spatial neighbours. This supposes a significant loss of samples that, together with the scarcity of the samples, can make the network result in overfitting. In order to avoid this problem, we have implemented a simple algorithm to replicate borders that allows us to feed the net with all the border pixels and to use them as any other pixel in the image. In this way, we can classify the complete hyperspectral image by replicating the pixels near the border, i.e. mirroring the $d/2$ pixels of border outwards, in order to create the corresponding patches or windows of the original border pixels, as illustrated graphically in Fig. 4 using the well-known AVIRIS Indian Pines hyperspectral image (described in detail in Section 4).

3.2. CNN architecture details

Once the edges are replicated and the hyperspectral image is split into 3-D patches, these are grouped in batches of size b and sent to the CNN. Then, the $d \times d \times n$ patches are sent as input volume to the first convolution layer ($c1$), composed by k^{c1} filters of $l^{c1} \times l^{c1} \times q^{c1}$, where $q^{c1} = n$, the stride is fixed to 1, and there is no padding.

After applying the ReLU function, the k^{c1} feature maps generated by $c1$ are sent to the first MaxPool layer ($mp1$), with a $l^{mp1} \times l^{mp1}$ kernel, stride of 2, and padding. The resulting output volume $p^{mp1} = d^{mp1} \times d^{mp1} \times k^{c1}$ is sent to the second convolution layer ($c2$) with k^{c2} filters of size $l^{c2} \times l^{c2} \times q^{c2}$, where $q^{c2} = k^{c1}$, with the same stride as in the first convolution and no padding either.

Again, after applying the ReLU function, the k^{c2} feature maps generated by $c2$ are sent to the second MaxPool layer ($mp2$), with a $l^{mp2} \times l^{mp2}$ kernel, stride of 2 and padding. The resulting output volume $p^{mp2} = d^{mp2} \times d^{mp2} \times k^{c2}$ is sent to the last convolution layer ($c3$), that has k^{c3} filters of size $l^{c3} \times l^{c3} \times q^{c3}$. This layer has the purpose of further refining the feature maps by processing each element one by one, so $k^{c3} = q^{c3} = k^{c2}$ and $l^{c3} = 1$. There is no third maxpool layer, so the output volume is reshaped in order to send it to fully-connected layers.

Four fully-connected layers were implemented ($fc1, fc2, fc3$ and $fc4$) with $l^{fc1}, l^{fc2}, l^{fc3}$ and l^{fc4} nodes, respectively. The first three fully-connected layers compute their output as $y^{fc} = f(w^{fc} y^{fc-1} + b^{fc})$, where w^{fc} are their weight matrices, b^{fc} are their bias vectors, y^{fc-1} is the output of the previous layer (in the first case, $y^{fc-1} = p^{C3}$, i.e. the output of $C3$ layer) and the activation function $f(\cdot)$ is ReLU. Finally, the last resulting matrix y^{fc3} is sent to $fc4$, which computes the outputs of the network with a softmax function as $y^{fc4} = w^{fc4} y^{fc3} + b^{fc4}$, where y^{fc4} contains the desired labels for the original $d \times d \times n$ input data.

In Algorithm 1 we can see a scheme of the operation of the proposed method. As we can notice, the method uses *cross-entropy* in order to determine the *loss* of the CNN model. It is defined as $H_{y'}(y) = \sum_i y'_i log(y_i)$, where y is our predicted probability distribution and y' is the true distribution, so the cross-entropy is a measure of how inefficiently predictions are calculated for describing the truth.

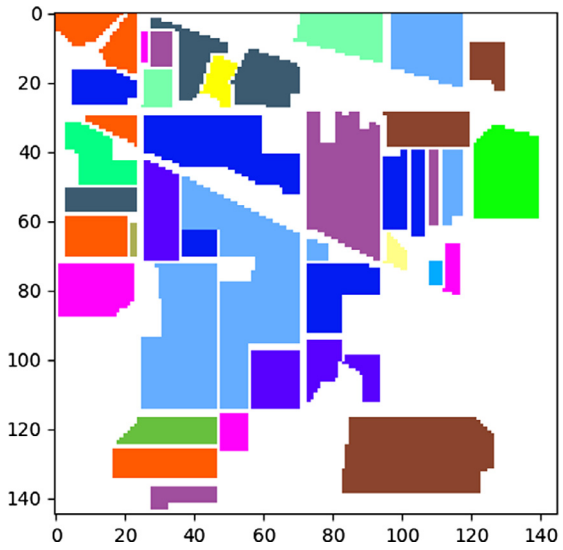
Algorithm 1. Proposed CNN method

```
procedure CNN_method(Y → original hyperspectral image)
    max_epochs → Set number of epochs value
    max_iters → Set number of iterations value
    d → Set patch size value
    n = Y.bands
    Y_norm = band_mean_normalize(Y)
    Y' = border_mirroring(Y_norm, d) → mirroring of d border pixels
    P = patches_creation(Y', d, n) → splitting Y' into patches of d × d × n
    T_a, T_b = training_test_sets(P) → training T_a and testing T_b sets
    G = batches_creation(T_a, b) → grouping patches in batches of size b
    for e < max_epochs do
        for it < max_iters do
            G' = get_next_batch(G)
            labels_G'_predicted = forward_pass(G')
            error = cross_entropy(labels_G', labels_G'_predicted)
            W, B = optimizer(error)
        end for
    end for
end procedure
```

As an interesting point we note that, for the initialization of all weights and bias of the network, we have used the so-called Xavier initializer (Glorot and Bengio, 2010), that allows the network to achieve greater stability, and the Adagrad optimizer (Duchi et al., 2011), as a simple method for learning rate adaptation. All the network characteristics (weights and bias initialization, optimizer, learning rate, steps, kernels size, use of padding and size of strides) are configurable through a JavaScript Object Notation (JSON)[9] file, as well as parameters d and b, which makes the implementation of the network flexible and easy to modify. Such implementation has been performed in Python.[10] In this way, the CNN takes advantage of the information of the central pixel neighbors as well as all the available spectral information, being able to adapt its structure easily and quickly.

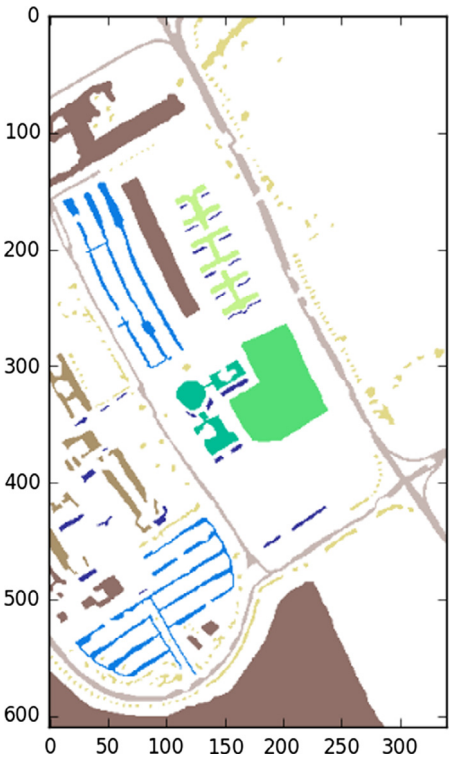
As we anticipated in Section 3.1, this kind of networks may suffer from an overfitting problem because of the complexity of the model derived from the large number of parameters that must be learned, and a lack of training samples (that is quite common in remote sensing applications). This problem may result in poor predictive performance in the testing phase, despite a high accuracy can be obtained in the training phase. To avoid such

[9] http://www.json.org/.
[10] https://www.python.org/.



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
	Soybeans-clean	693
	Wheat	205
	Woods	1265
	Bldg-Grass-Tree-Drives	386
	Stone-steel towers	93
Total samples		21025

Fig. 5. Original ground-truth image of the AVIRIS Indian Pines scene and number of samples per class.



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

Fig. 6. Original ground-truth image of the ROSIS University of Pavia scene and number of samples per class.

overfitting problems, we have allowed an optional and configurable dropout mechanism in the first and second convolution layers. The dropout method sets the output of some randomly selected hidden neurons to zero, so that the dropped neurons do not contribute in the forward pass and they are not used in the back-propagation stage (Hinton et al., 2012).

To conclude this section, it is important to note that the proposed CNN has been implemented using the TensorFlow open source library for machine intelligence[11] including its GPU functionalities, that allow for fast performance even when dealing with very large hyperspectral image volumes. In the following section, we evaluate the proposed CNN from the viewpoint of both computational performance and classification accuracy.

4. Experiments and results

4.1. Experimental configuration

In order to evaluate the performance of our newly presented CNN architecture, we use a hardware environment composed by

[11] https://www.tensorflow.org.

a 6th Generation Intel® Core™ i7-6700 K processor with 8 M of Cache and up to 4.20 GHz (4 cores/8 way multitask processing), 40 GB of DDR4 RAM with a serial speed of 2400 MHz, a GPU NVIDIA GeForce GTX 1080 with 8 GB GDDR5X of video memory and 10Gbps of memory frequency, a Toshiba DT01ACA HDD with 7200RPM and 2 TB of capacity, and an ASUS Z170 pro-gaming motherboard.

4.2. Hyperspectral image data

In our experiments, we have used two well-known hyperspectral image data sets. The first one is the Indian Pines image, gathered in 1992 by the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) sensor (Green et al., 1998) over a set of agricultural fields with regular geometry and with a multiple crops and

Table 1
Indian Pines: number of samples in the training set used by the proposed method and by the method in Chen et al. (2016).

Indian Pines					
clase	pixels	200 samples per class	100 samples per class	50 samples per class	Chen et al. (2016)
Alfalfa	46	33	33	33	30
Corn-notill	1428	200	100	50	150
Corn-min	830	200	100	50	150
Corn	237	181	100	50	100
Grass/pasture	483	200	100	50	150
Grass/trees	730	200	100	50	150
Grass/pasture-mowed	28	20	20	20	20
Hay-windrowed	478	200	100	50	150
Oats	20	14	14	14	15
Soybeans-notill	972	200	100	50	150
Soybeans-min	2455	200	100	50	150
Soybeans-clean	593	200	100	50	150
Wheat	205	143	100	50	150
Woods	1265	200	100	50	150
Bldg-grass-tree-drives	386	200	100	50	50
Stone-steel towers	93	75	75	50	50
Total	10249	2466	1342	717	1765

Table 2
University of Pavia: number of samples in the training set used by the proposed method and by the method in Chen et al. (2016).

Pavia University					
class	pixels	200 samples per class	100 samples per class	50 samples per class	Chen et al. (2016)
Asphalt	6631	200	100	50	548
Meadows	18649	200	100	50	540
Gravel	2099	200	100	50	392
Trees	3064	200	100	50	542
Painted metal sheets	1345	200	100	50	256
Bare soil	5029	200	100	50	532
Bitumen	1330	200	100	50	375
Self-blocking bricks	3682	200	100	50	514
Shadows	947	200	100	50	231
Total	42776	1800	900	450	3930

Table 3
Configuration of the CNN architecture for the Indian Pines and University of Pavia datasets. The kernel size (like the number and type of layers, strides and padding) is one of the design choices of the proposed CNN architecture. Large kernels allow our CNN to learn more complex features, although with larger kernels the computational time of the training/testing phase is also greater.

CNN proposed topologies							
Hyperspectral datasets	convolution layers				Fully Connected Layers		
	Kernel size $k^c \times l^c \times l^c \times q^c$	*ReLU*	*Pooling* $l^{mp} \times l^{mp}$	*Dropout*	*N°neurons* l^{fc}	*Function*	*Dropout*
Indian Pines	$600 \times 5 \times 5 \times 200$	Yes	2×2	Yes (10%)	1024	ReLU	Yes(10%)
	$200 \times 3 \times 3 \times 600$	Yes	2×2	Yes (10%)	1024	ReLU	No
	$200 \times 1 \times 1 \times 200$	Yes	No	No	512	ReLU	No
					16	Softmax	No
Pavia University	$380 \times 7 \times 7 \times 103$	Yes	2×2	Yes(20%)	2048	ReLU	No
	$350 \times 5 \times 5 \times 380$	Yes	2×2	Yes(20%)	2048	ReLU	No
	$350 \times 1 \times 1 \times 350$	Yes	No	No	1024	ReLU	No
					9	Softmax	No
Common parameters							
Bach size *b*	Steps (iterations)	Epochs	Learning rate	Optimizer			
100	1500	20	0.01	AdagradOptimizer			

irregular patches of forest in Northwestern Indiana. The AVIRIS Indian Pines scene has 145 × 145 pixels with 224 spectral bands in the range from 400 to 2500 nm, with 10 nm of spectral resolution, 20 m moderate spatial resolution, and 16 bits radiometric resolution. After an initial analysis, 4 zero bands and another 20 bands with lower signal-to-noise ratio (SNR) have been removed because

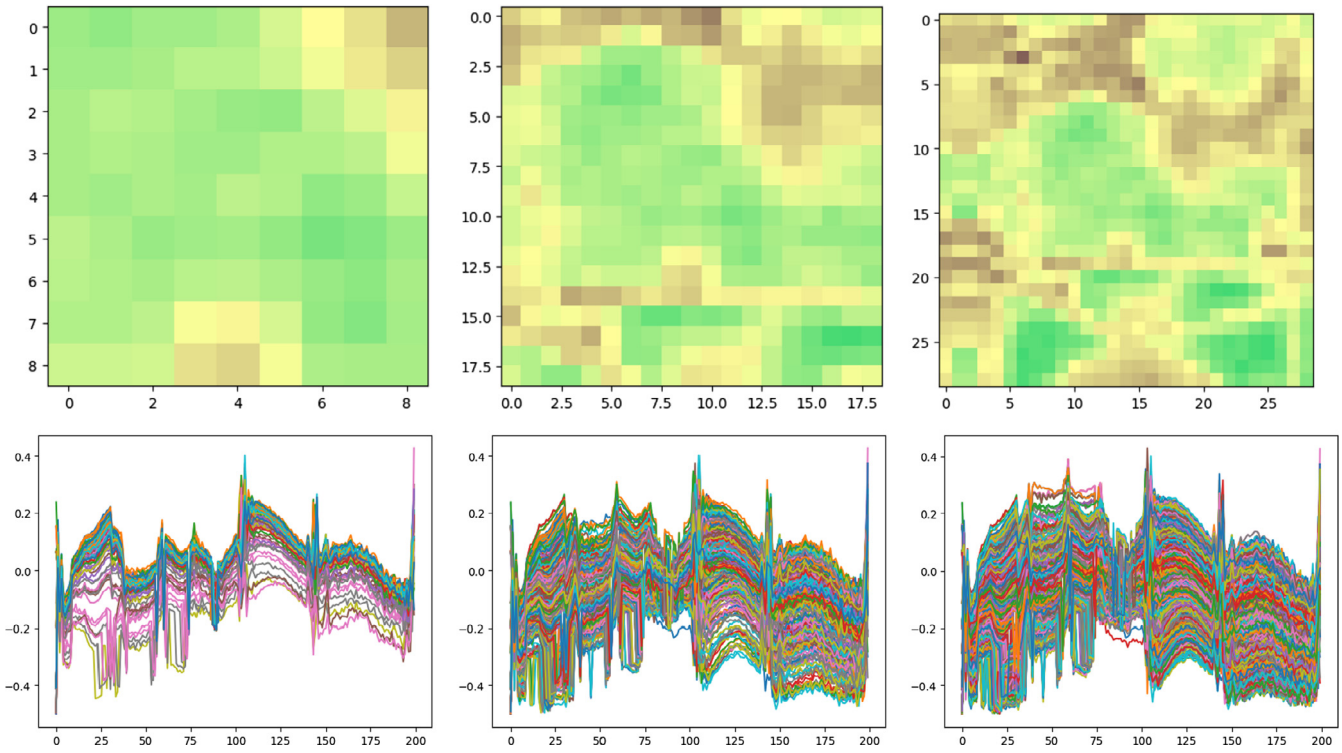


Fig. 7. Illustration of the effect of using different patch sizes for the Indian Pines scene. The upper row shows the size of the neighborhood around the central pixel when we use $d = 9, d = 19$ and $d = 29$, as we can see on the axes from left to right (color map of band 140). In the lower row we can see the spectral signature of the pixels that form each neighborhood, respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

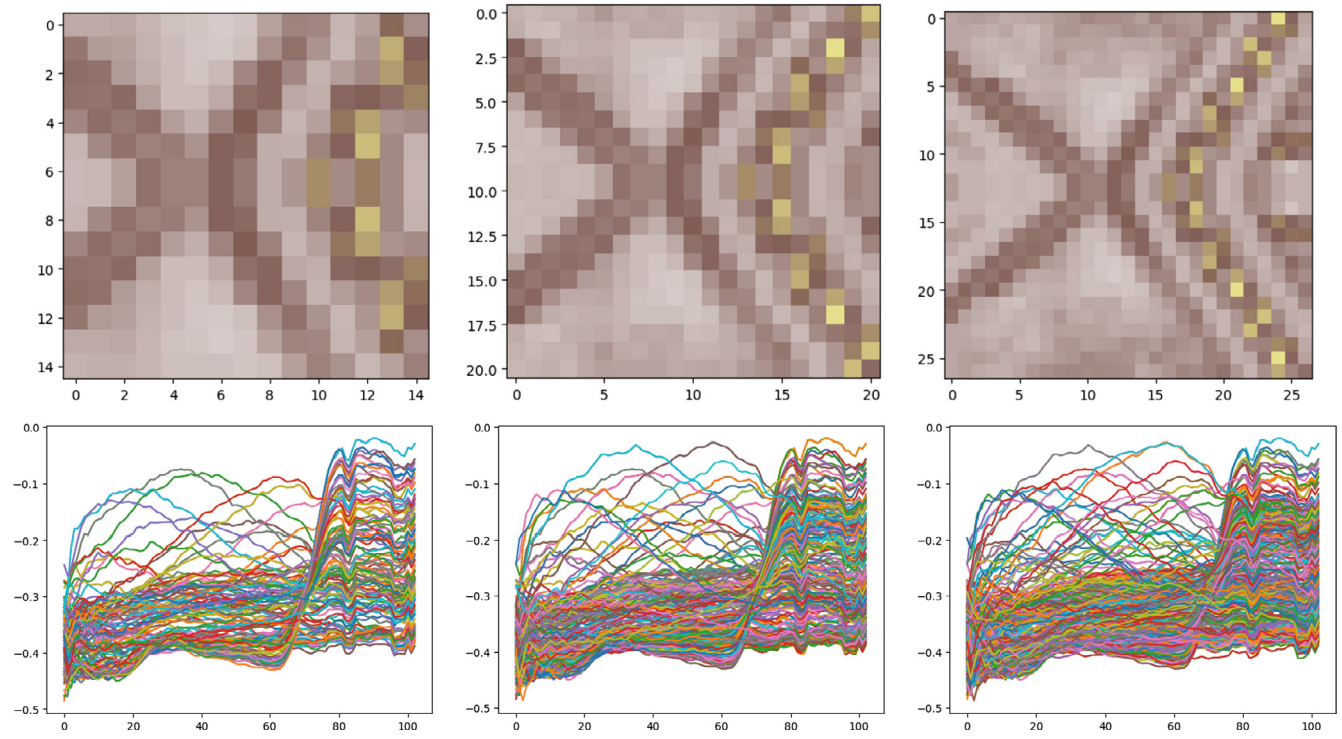


Fig. 8. Illustration of the effect of using different patch sizes for the University of Pavia scene. The upper row shows the size of the neighborhood around the central pixel when we use $d = 15, d = 21$ and $d = 27$, as we can see on the axes from left to right (color map of band 10). In the lower row we can see the spectral signature of the pixels that form each neighborhood, respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

of atmospheric absorption phenomena in those bands, retaining only 200 spectral channels. Moreover, about half of the pixels in the hyperspectral image (10249 of 21025, i.e. 48.74%) contain ground-truth information, which comes in the form of a single label assignment for each pixel with a total of 16 ground-truth classes (see Fig. 5).

The second data set used in experiments was collected by the Reflective Optics System Imaging Spectrometer (ROSIS) sensor

Table 4
Execution times (in seconds) and accuracies measured with patches of size $d = 9, d = 19$ and $d = 29$ for the Indian Pines dataset and $d = 15, d = 21$ and $d = 27$ for the University of Pavia dataset. The CNN configuration is the one indicated in Table 3, with 1500 iterations, 100 samples per class and 5 executions.

Dataset	Patch size	Time per step		Total time		Accuracy	
		Avg.	Std. dev.	Avg.	Std. dev.	Avg.	Std. dev.
Indian Pines	$d = 9$	0.02	0.01	29.22	1.05	78.46	4.45
	$d = 19$	0.08	0.02	116.30	3.22	91.34	1.59
	$d = 29$	0.16	0.03	248.29	7.91	95.53	0.48
University of Pavia	$d = 15$	0.02	0.01	34.78	1.13	94.02	0.62
	$d = 21$	0.05	0.02	74.66	2.22	95.08	1.41
	$d = 27$	0.09	0.02	131.66	2.88	94.13	0.66

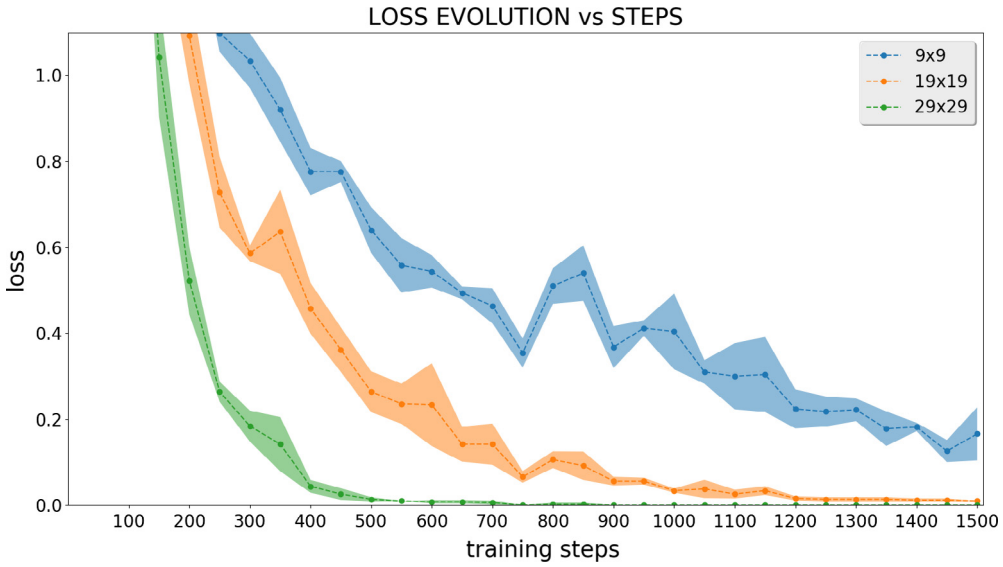


Fig. 9. Indian Pines: Evolution of the validation loss in terms of steps (iterations) with $d = 9, d = 19$ and $d = 29$. The shadow shows the standard deviation of the loss for the five executions of each patch (zoom in error interval $[0, 1]$). This experiment has been executed using 1500 iterations.

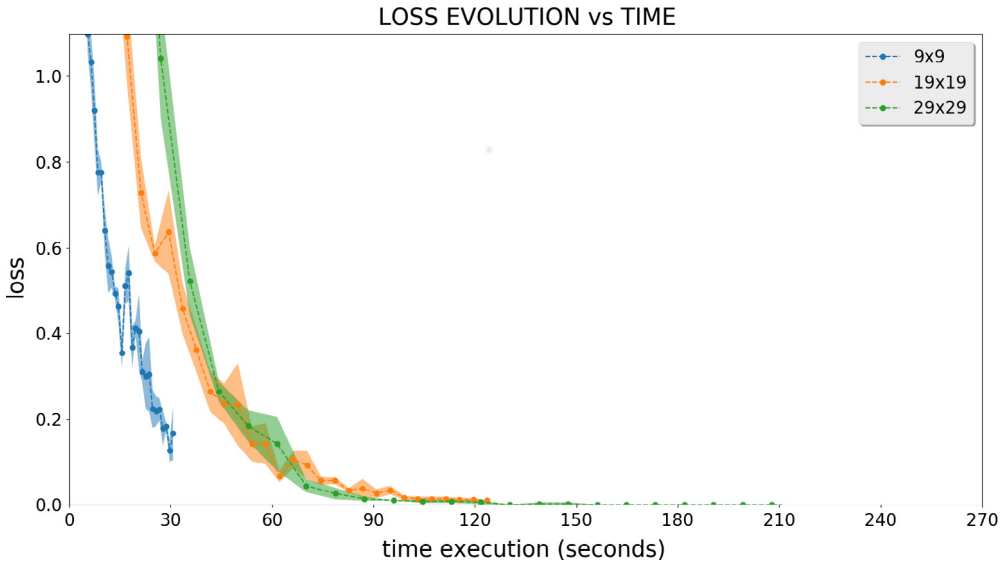


Fig. 10. Indian Pines: Evolution of the validation loss in terms of time (seconds) with $d = 9, d = 19$ and $d = 29$. The shadow shows the standard deviation of the loss for the five executions of each patch (zoom in error interval $[0, 1]$). This experiment has been executed using 1500 iterations.

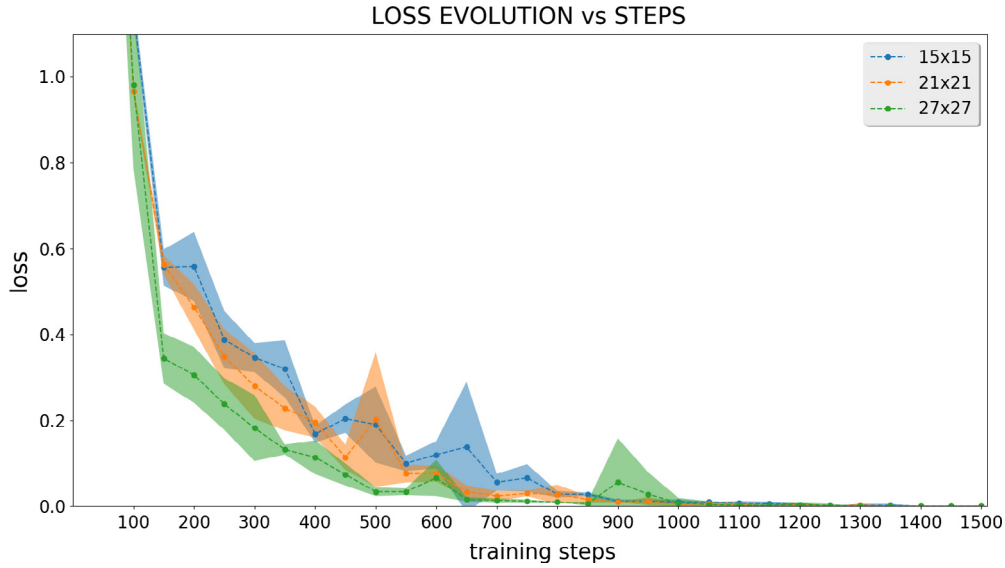


Fig. 11. Pavia University: Evolution of the validation loss in terms of steps (iterations) with $d = 15, d = 21$ and $d = 27$. The shadow shows the standard deviation of the loss for the five executions of each patch (zoom in error interval $[0, 1]$). This experiment has been executed using 1500 iterations.

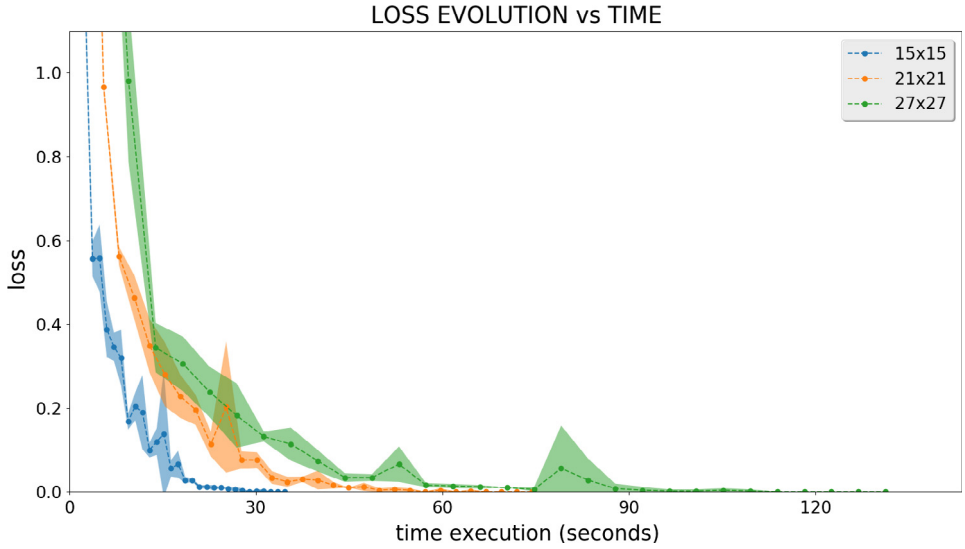


Fig. 12. Pavia University: Evolution of the validation loss in terms of time (seconds) with $d = 15, d = 21$ and $d = 27$. The shadow shows the standard deviation of the loss for the five executions of each patch (zoom in error interval $[0, 1]$). This experiment has been executed using 1500 iterations.

(Kunkel et al., 1988) during a flight campaign over the city of Pavia, in northern Italy. The dataset covers an urban environment, with various solid structures (asphalt, gravel, metal sheets, bitumen, bricks), natural objects (trees, meadows, soil), and shadows (9 classes in total). Other objects whose compositions differ from the labeled ones are considered as clutter. The scene was collected over an university area. It contains 103 spectral bands with 610×340 pixels in the spectral range from 0.43 to 0.86 μm, and spatial resolution of 1.3 m/pixel. About 20.62% of the pixels in the hyperspectral image (42776 of 207400) contain ground-truth information (see Fig. 6).

4.2.1. Data preprocessing: division of training and testing sets

When our method divides the hyperspectral images into training and testing sets, it follows a method of preprocessing with class balancing that must be conveniently explained at this point. In addition to their huge dimensionality and the existing correlation between the spectral features collected (Melgani and Bruzzone, 2004), hyperspectral images present another complication: the class imbalance problem (He and Garcia, 2009). This problem appears in a dataset when some of the classes are heavily underrepresented (in terms of their labeled samples) as compared to other classes (García et al., 2011), leading to poor classification performance in many real-world applications, especially for the minority classes. In order to deal with this problem, and taking into account that we could not identify a common pattern about sample selection strategies in the literature, we have tried to keep a stratified sampling strategy in our experiments. As a result, we tried to balance the number of samples selected in accordance with the number of available samples per class.

The first step is to divide randomly the original dataset in two subsets: the first one (training set) with 75%[12] of the samples and the second one (testing set) with the remaining 25%. If we keep that

[12] For each class we make sure that about 75% of each class is taken, until we have a number of samples close to 75% of the complete dataset, so we make sure that all classes are represented in the subset.

75% of samples for the training set, we will once again experience the class imbalance problem. For instance, in the Indian Pines image the Alfalfa class has only 46 samples as opposed to Corn-notill, which has 1428. 75% of each is $34,5 \approx 36$ for Alfalfa and 1071 for Corn-notill, a completely unbalanced result. The solution adopted in this case is to simply reduce the number samples until a balanced

Table 5
Classification accuracies obtained by our CNN (with patch sizes of $d = 9, d = 19$ and $d = 29$) for the Indian Pines hyperspectral dataset. We used 2500 iterations and repeated the experiment 5 times.

Neural networks	CNN $d = 9$			CNN $d = 19$			CNN $d = 29$		
Samples per class	50	100	200	50	100	200	50	100	200
Alfalfa	98.70 (1.06)	99.13 (1.06)	99.13 (1.06)	99.57 (0.87)	100.00 (0.00)	99.57 (0.87)	99.13 (1.74)	99.57 (0.87)	99.13 (1.06)
Corn-notill	70.76 (3.42)	76.93 (2.50)	80.48 (9.37)	76.74 (3.89)	85.84 (1.92)	94.47 (2.46)	82.10 (3.94)	91.32 (0.46)	98.17 (0.67)
Corn-min	78.92 (5.37)	90.55 (1.97)	96.65 (1.20)	82.77 (2.61)	93.23 (2.09)	98.22 (0.87)	86.41 (4.29)	94.84 (0.47)	98.92 (0.68)
Corn	96.54 (1.89)	99.75 (0.21)	99.66 (0.17)	99.41 (0.83)	99.66 (0.32)	100.00 (0.00)	97.81 (2.92)	100.00 (0.00)	100.00 (0.00)
Grass/Pasture	89.86 (4.56)	97.81 (0.81)	99.46 (0.52)	95.20 (1.54)	98.18 (0.73)	99.75 (0.20)	96.15 (3.75)	98.34 (1.23)	99.71 (0.21)
Grass/Trees	97.40 (0.94)	98.11 (0.88)	99.53 (0.32)	92.96 (2.64)	97.92 (1.31)	98.90 (0.43)	96.47 (2.43)	98.66 (0.94)	99.40 (0.52)
Grass/pasture-mowed	100.00 (0.00)	100.00 (0.00)	100.00 (0.00)	100.00 (0.00)	100.00 (0.00)	100.00 (0.00)	100.00 (0.00)	100.00 (0.00)	100.00 (0.00)
Hay-windrowed	99.29 (0.34)	99.62 (0.08)	99.67 (0.21)	99.12 (1.06)	99.08 (0.45)	99.62 (0.47)	99.62 (0.50)	99.96 (0.08)	100.00 (0.00)
Oats	100.00 (0.00)	100.00 (0.00)	100.00 (0.00)	100.00 (0.00)	100.00 (0.00)	100.00 (0.00)	100.00 (0.00)	100.00 (0.00)	100.00 (0.00)
Soybeans-notill	80.39 (3.52)	87.86 (1.95)	92.43 (2.03)	88.35 (4.26)	94.81 (1.94)	98.00 (0.78)	88.66 (1.62)	95.21 (1.26)	98.62 (1.39)
Soybeans-min	65.82 (5.83)	79.45 (1.74)	76.42 (4.61)	69.52 (3.63)	85.85 (1.35)	94.32 (2.02)	79.40 (1.33)	90.52 (1.16)	96.15 (0.57)
Soybean-clean	80.84 (4.29)	90.96 (3.58)	97.74 (0.86)	84.65 (3.19)	96.90 (1.61)	99.09 (0.45)	88.67 (1.80)	97.17 (1.64)	99.33 (0.18)
Wheat	99.61 (0.78)	99.80 (0.39)	99.71 (0.24)	99.90 (0.20)	99.90 (0.20)	100.00 (0.00)	99.32 (0.39)	100.00 (0.00)	99.90 (0.20)
Woods	91.21 (1.17)	94.80 (1.43)	97.71 (0.74)	88.35 (3.57)	95.54 (1.43)	98.85 (0.94)	96.74 (1.22)	98.12 (0.54)	98.96 (0.46)
Bldg-Grass-Tree-Drives	91.14 (2.07)	98.50 (0.81)	99.27 (0.60)	96.94 (3.11)	98.81 (1.42)	99.90 (0.13)	99.07 (0.26)	99.69 (0.25)	100.00 (0.00)
Stone-steel towers	100.00 (0.00)	100.00 (0.00)	100.00 (0.00)	99.57 (0.86)	100.00 (0.00)	100.00 (0.00)	99.78 (0.43)	98.92 (0.96)	100.00 (0.00)
Overall Accuracy	80.85 (1.58)	88.46 (0.32)	90.11 (0.67)	83.73 (1.30)	92.54 (0.16)	97.23 (0.30)	88.78 (0.78)	95.05 (0.28)	98.37 (0.17)
Average Accuracy	90.03 (0.98)	94.58 (0.22)	96.12 (0.28)	92.07 (0.66)	96.61 (0.24)	98.79 (0.10)	94.33 (0.35)	97.64 (0.13)	99.27 (0.11)
Kappa	78.44 (1.74)	86.92 (0.36)	88.81 (0.76)	81.68 (1.44)	91.54 (0.18)	96.85 (0.34)	87.31 (0.88)	94.38 (0.31)	98.15 (0.19)
Run time	47.97 (0.01)	48.50 (0.01)	48.21 (0.01)	193.40 (0.01)	192.85 (0.01)	197.65 (0.01)	421.07 (0.02)	404.15 (0.02)	405.46 (0.02)

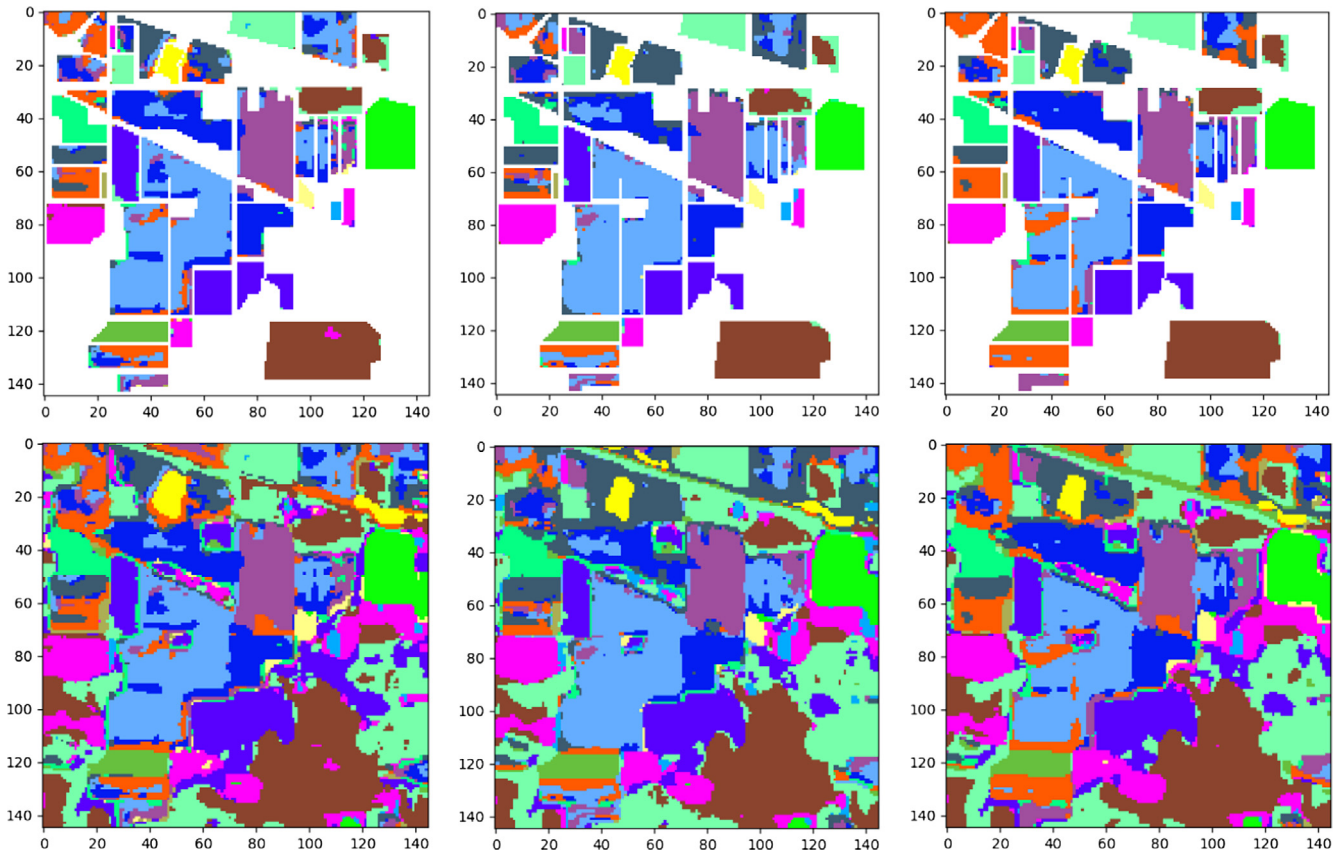


Fig. 13. Classification results for Indian Pines image with $d = 9$ and 50 (left), 100 (center) and 200 (right) samples per class. The upper row displays the classification result without the background, and the lower row displays the classification result with the background.

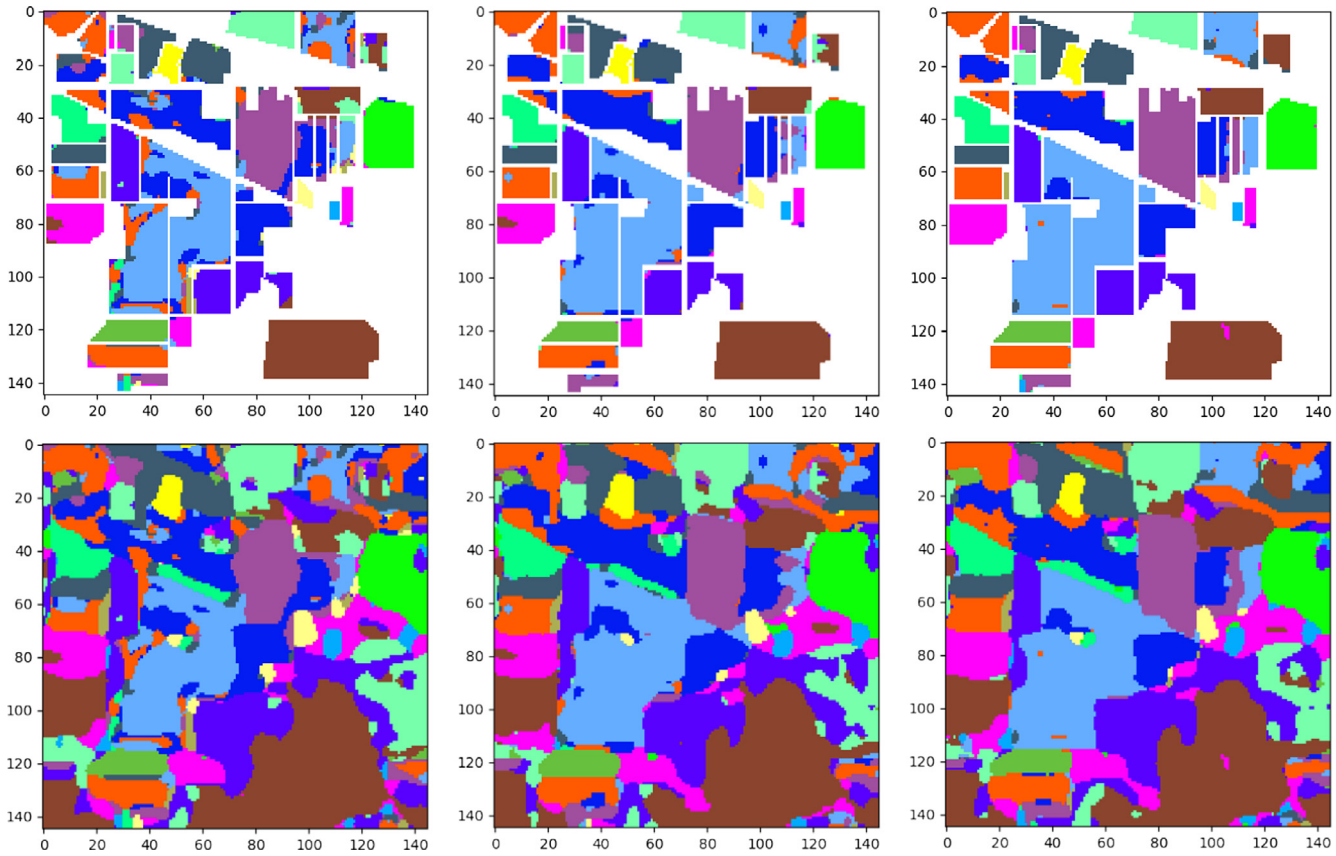


Fig. 14. Classification results for Indian Pines image with $d = 19$ and 50 (left), 100 (center) and 200 (right) samples per class. The upper row displays the classification result without the background, and the lower row displays the classification result with the background.

result is achieved. After we get 75% sampling of each class, we set a maximum number of samples per class (as a threshold), for example 50, 100 or 200 samples per class. For those classes with many samples, we simply cut the samples until reaching the threshold. However, for those classes that have very few samples and do not reach the threshold, only those available pixels are taken. In Table 1 we can observe the real number of samples that we are using in each experiment when we refer to "50 samples per class", "100 samples per class" or "200 samples per class". Except for those classes that do not reach the proposed threshold (and that use 75% complete), the rest of classes work with 15–25% of their samples. In Table 2 we can see that the same solution is adopted for the University of Pavia data. In both cases, the proposed method uses less samples than (Chen et al., 2016) with the exception of Indian Pines with 200 samples per class.

4.3. Hyperparameter tuning

The first step to carry out the experiments has been to adjust the configuration parameters of the convolutional network to get the best possible classification accuracy in the considered hyperspectral datasets, through cross-validation.

For the Indian Pines dataset, the CNN configuration parameters have been adjusted according to Table 3. As we can see in the convolution layers, the noise of some of the Indian Pines image bands is mitigated by a first expansion of depth, and the overfitting problem is solved by adding dropout in the first and second convolution layers and in the first fully connected layer.[13] The third convolution layer refines the $c2$'s feature maps with the objective of obtaining a better classification.

For the University of Pavia dataset, the CNN configuration parameters have been adjusted according to Table 3. Also, we improved the quality of the spectral information by extending the depth of the feature maps in the first convolution. In this case, the overfitting problem is worse than in the Indian Pines scene (mainly due to the greater number of parameters to be learned), so we increased the value of the dropout in the first and second convolution layers. Again, the third convolution layer refines the $c2$'s feature maps, with the objective of obtaining a better classification.

4.4. Performance evaluation

To test the proposed CNN for hyperspectral image classification with the configurations described in Section 4.3, several experiments have been conducted, first with the Indian Pines hyperspectral dataset and, second, with the University of Pavia hyperspectral dataset. At this point, we emphasize that data pre-processing plays a very important role in this kind of deep learning algorithms. In practice, many classification methods work better after a data normalization procedure. In this case hyperspectral datasets have been scaled between in the range $[-0.5, 0.5]$ and a band-mean normalized procedure has been performed. This means that each of the spectral channels in the image have been normalized by subtracting the mean.

Testing parameter d: We tested different sizes of parameter d, using a fixed number of 100 samples per class. For the Indian Pines data, we have considered three patch sizes: $d = 9, d = 19$ and $d = 29$. In Fig. 7 we illustrate the effect of using different patch sizes over a random pixel in the Indian Pines dataset. As we add

[13] After the first experiment, we realized that these values were insufficient and the network was fine-tuned again. A 10% dropout was added to the third convolution layer and a 30% dropout was added to the first fully-connected layer to avoid overfitting.

pixels to the neighborhood, the spatial information around the considered pixel is more clear, especially when it comes to edge pixels. However, a patch too large can detract from the target pixel. As for the spectral signature, adding more neighbors to the target pixel also makes the signature as a whole more defined.

For the Pavia University data, we have tested other patch sizes: $d = 15, d = 21$ and $d = 27$. The difference is motivated by our pre-assessment of the size of relevant features in the image. We also provide in Fig. 8 an illustrative example of how the size of patches impacts the overall performance in the University of Pavia data set. In this case, the scene presents many object borders in the leftmost part. By adding more spatial information we can better identify the pixels belonging to such edges. However, given the reduced number of classes, if we add too many spectral signatures we can make the patches slightly homogeneous.

For each d, we divided the original hyperspectral image into pieces (mirroring the borders of the image, if necessary) and grouping the patches into training samples and test samples. To split the patches, we followed the steps described in Section 4.2.1. Each execution of this experiment has been repeated 5 times.

Table 4 reports the obtained results. We can observe that, for the Indian Pines image, $d = 29$ achieves the best result, reaching an overall accuracy of 95.53% with a smaller error in fewer iterations (with only 400 iterations, $d = 29$ has already reached a

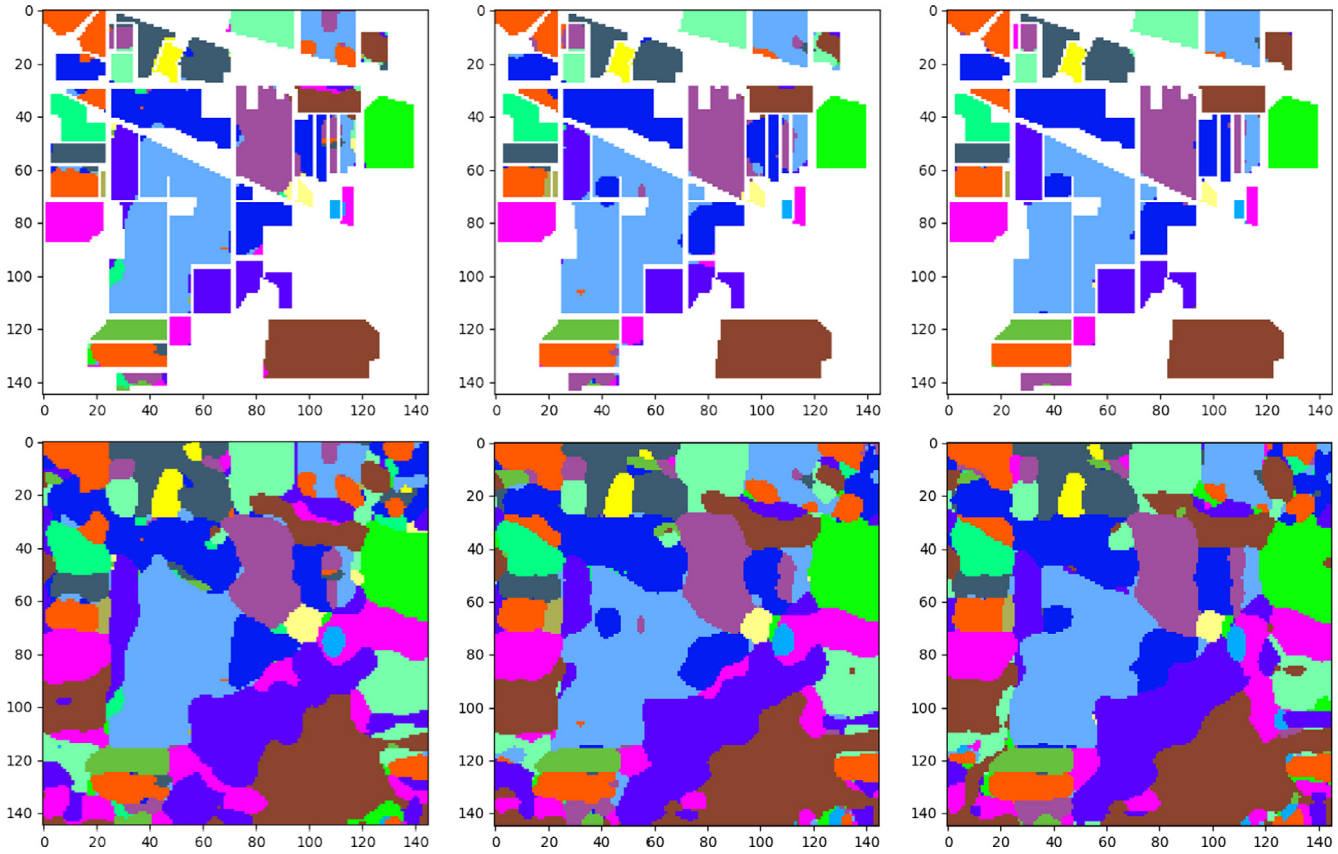


Fig. 15. Classification results for Indian Pines image with $d = 29$ and 50 (left), 100 (center) and 200 (right) samples per class. The upper row displays the classification result without the background, and the lower row displays the classification result with the background.

Table 6
Obtained classification accuracies (with patch sizes of $d = 15, d = 21$ and $d = 27$) for the University of Pavia hyperspectral dataset. We used 1500 iterations and repeated the experiment 5 times.

Neural networks	CNN $d = 15$			CNN $d = 21$			CNN $d = 27$		
Samples per class	50	100	200	50	100	200	50	100	200
Asphalt	79.86 (4.46)	90.58 (0.74)	92.81 (0.72)	86.41 (0.47)	91.30 (1.81)	95.31 (0.97)	82.66 (1.18)	92.07 (1.05)	96.31 (0.19)
Meadows	88.97 (1.65)	94.20 (1.94)	97.20 (0.95)	89.44 (4.85)	93.39 (1.52)	98.16 (0.12)	90.39 (3.03)	93.56 (1.76)	97.54 (0.39)
Gravel	83.52 (1.92)	92.28 (2.12)	96.97 (0.90)	85.31 (4.51)	92.01 (3.55)	97.92 (0.59)	88.74 (0.90)	93.66 (1.16)	96.84 (0.29)
Trees	96.31 (1.02)	97.45 (1.34)	98.62 (0.43)	94.36 (0.49)	96.87 (0.74)	98.74 (0.18)	90.88 (1.35)	94.51 (2.12)	97.58 (0.41)
Painted metal sheets	99.83 (0.09)	99.93 (0.11)	100.00 (0.00)	99.38 (0.13)	99.88 (0.09)	100.00 (0.00)	99.31 (0.21)	99.53 (0.37)	99.65 (0.15)
Bare Soil	90.72 (1.47)	95.18 (0.93)	98.57 (0.74)	93.54 (2.45)	98.50 (1.20)	99.57 (0.31)	88.73 (2.00)	97.67 (0.88)	99.33 (0.25)
Bitumen	91.88 (1.61)	92.38 (1.22)	97.27 (0.96)	91.00 (0.57)	96.19 (1.50)	99.75 (0.09)	92.73 (1.25)	95.16 (1.29)	98.90 (1.14)
Self-Blocking Bricks	82.91 (5.26)	92.07 (1.49)	96.17 (1.70)	89.77 (2.62)	94.41 (0.93)	98.20 (0.21)	91.74 (1.49)	94.88 (1.43)	98.89 (0.47)
Shadows	99.65 (0.22)	99.54 (0.51)	99.86 (0.13)	99.65 (0.10)	99.75 (0.13)	99.82 (0.18)	97.40 (1.87)	98.91 (0.88)	99.58 (0.09)
Overall Accuracy	88.17 (0.31)	93.95 (0.74)	96.83 (0.19)	90.22 (1.78)	94.37 (1.10)	98.06 (0.13)	89.58 (1.95)	94.35 (1.05)	97.80 (0.22)
Average Accuracy	90.40 (0.38)	94.85 (0.41)	97.50 (0.14)	92.10 (0.63)	95.81 (0.82)	98.61 (0.09)	91.40 (1.26)	95.55 (0.68)	98.29 (0.25)
Kappa	84.63 (0.36)	92.07 (0.94)	95.83 (0.24)	87.28 (2.21)	92.63 (1.42)	97.44 (0.18)	86.37 (2.50)	92.61 (1.35)	97.09 (0.29)
Run time	43.02 (0.01)	42.78 (0.01)	42.83 (0.01)	74.30 (0.01)	74.17 (0.01)	74.09 (0.01)	132.77 (0.02)	132.48 (0.02)	132.68 (0.02)

minimum error, while for $d = 19$ it needs about 1000 iterations to reach the same error and $d = 9$ is not able to reduce its error in 1500 iterations, see Fig. 9). However, the time required for each step when $d = 29$ is adopted is greater than with $d = 19$ or $d = 9$: each step of $d = 29$ is 0.08 s slower than the steps of $d = 19$ and 0.14 s slower than the steps of $d = 9$ (see Fig. 10). In terms of the accuracy/time ratio, the best option is $d = 19$, although the best accuracy is achieved by using $d = 29$ (but its execution time is larger).

On the other hand, the results obtained for the University of Pavia dataset are also shown in Table 4. In this case, the patch with size $d = 21$ achieves the best accuracy results: 95.08% in 74.66 s, reaching 1.06 percentage more than $d = 15$ and 0.95 percentage more than $d = 27$. As for the number of iterations, in Fig. 11 we can see that $d = 27$ needs less iterations to reach an acceptable error (between 700–800 iterations) while $d = 21$ needs around 1000–1100 iterations, which is very similar to $d = 15$ that also needs around 1000–1100 iterations to reach a low error. On the other hand, the time per iteration for each patch size is different, being the fastest $d = 15$ (2.15 times faster than $d = 21$ and 3.79 times faster than $d = 27$) and the slowest $d = 27$ (around 1.76 times slower than $d = 21$), see Fig. 12. With this information at hand, we can conclude that the best patch size is $d = 21$, as it reaches the best result in a fairly reasonable time.

Testing the number of samples per class: At this point, we tested the accuracy achieved for different patch sizes

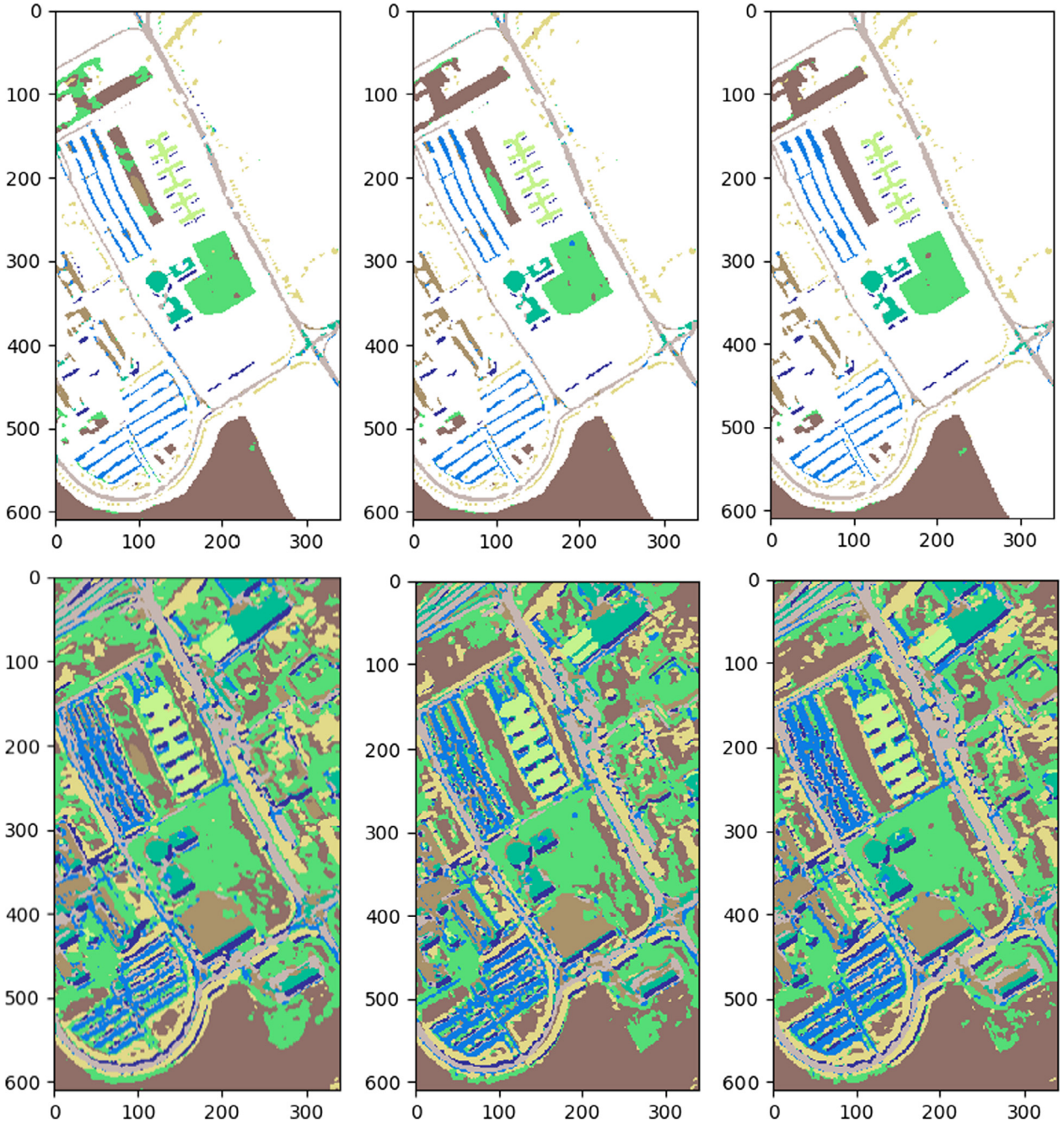


Fig. 16. Classification results for the University of Pavia data set with $d = 15$ and 50 (left), 100 (center) and 200 (right) samples per class. The upper row displays the classification result without the background, and the lower row displays the classification result with the background.

($d = 9, d = 19$ and $d = 29$ for Indian Pines and $d = 15, d = 21$ and $d = 27$ for Pavia) with different amounts of training data, in particular with 50, 100 and 200 samples per class.

The results obtained for the Indian Pines dataset are shown in Table 5 (with standard deviation). In this case, for each experiment the parameters of the CNN have been fine tuned, in order to achieve the best possible accuracy. As we can observe in Figs. 9 and 10, the configuration of the CNN in Table 3 for Indian Pines presents a marked overfitting problem (that the standard deviation also seems to indicate). As a result, the dropout percentages have been modified for the Indian Pines experiment, specifically we add a 10% dropout in the third convolution layer and raise the dropout from 10% to 30% on the first fully-connected layer. Also, for this experiment we have increased the number of iterations from 1500 to 2500. Thanks to the overfitting reduction, the network is able to converge much faster, drastically reducing the initial execution times (i.e., being 1.32 times faster with $d = 9$, 1.57 times faster with $d = 19$ and 1.66 times faster with $d = 29$ with 100 samples per class). In addition, stability has been improved by reducing the standard deviation of each run to 0.01. In Table 5 we can observe that the results with $d = 9, d = 19$ and $d = 29$ increase (in terms of accuracy) as more training samples per class are included, reaching the maximum values with 200 samples per class. Again, $d = 29$ reaches the best accuracy results with 200

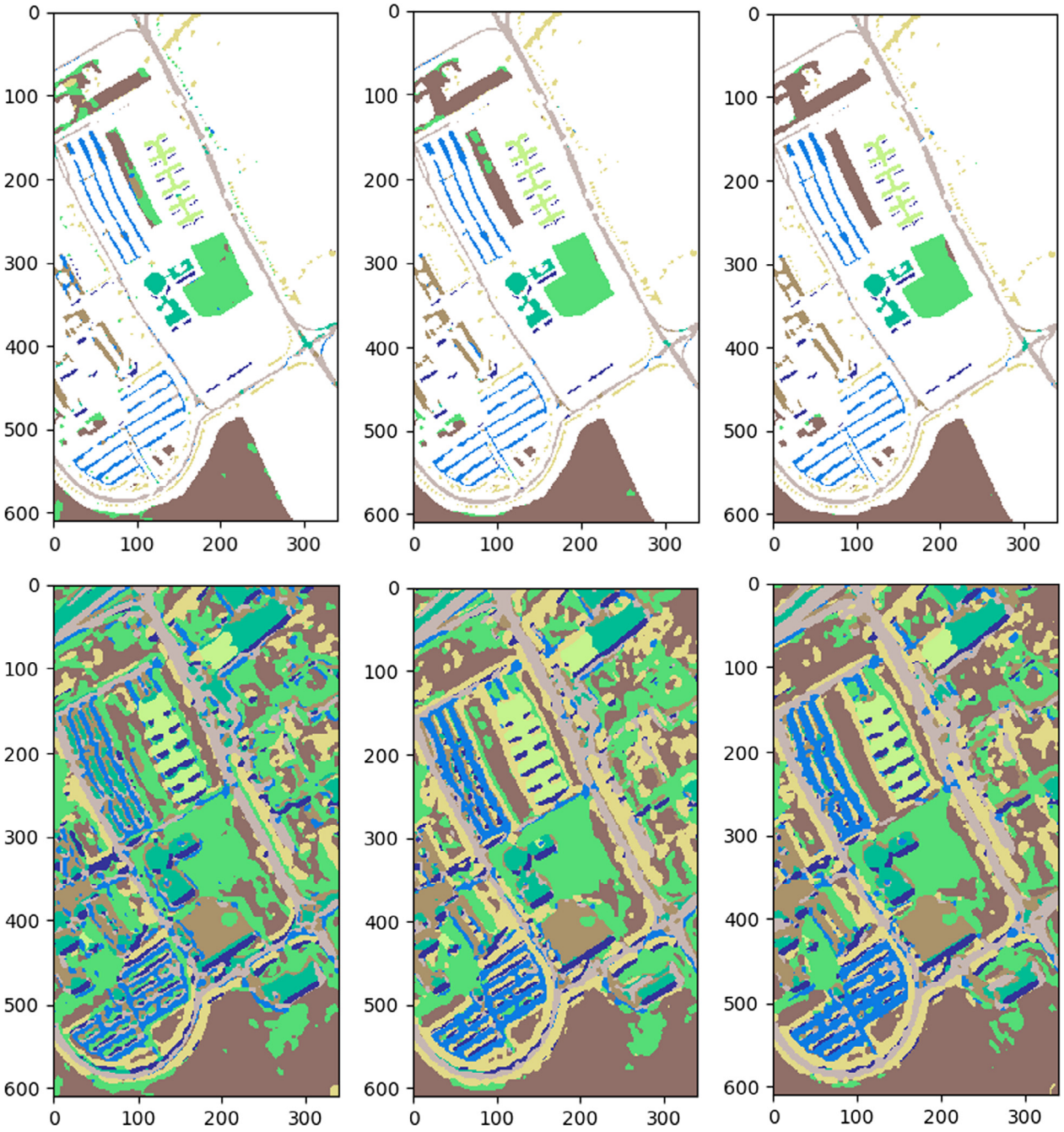


Fig. 17. Classification results for the University of Pavia data set with $d = 21$ and 50 (left), 100 (center) and 200 (right) samples per class. The upper row displays the classification result without the background, and the lower row displays the classification result with the background.

samples per class, although the results obtained with a patch size of $d = 19$ are quite similar to those found using $d = 29$, just one or two percentage points below.

In Figs. 13–15 we report the classification maps obtained in each experiment, without the mirroring of the borders. First, in Fig. 13 we can see the classified image without background (the first three images) and with background (the last three), obtained with a patch size of $d = 9$ and 50, 100 and 200 samples per class. When the background is removed, we can see how the pixels of each class are mixed, in particular near the edges. Also, with background the results are poorly defined, although these are improved by adding more training samples in each class.

Secondly, in Fig. 14 we can see the classified images (with and without background) obtained with a patch size of $d = 19$ and 50, 100 and 200 samples per class. With better results than with $d = 9$, the borders between the classes are better defined in this case, also in the classification with background. The best result is achieved with 200 samples per class.

Finally, in Fig. 15 we can observe the obtained classified images with a patch size of $d = 29$. First we report the classification images without background and below them, we show the corresponding ones with background. With only ground-truth pixels, the first three images show better defined classes than the previously displayed ones with $d = 9$ and $d = 19$, being the classification map obtained with 200 samples per class the most similar to the original ground-truth image. Even in the classification with background, classes appear better defined since the borders between them are more regular.

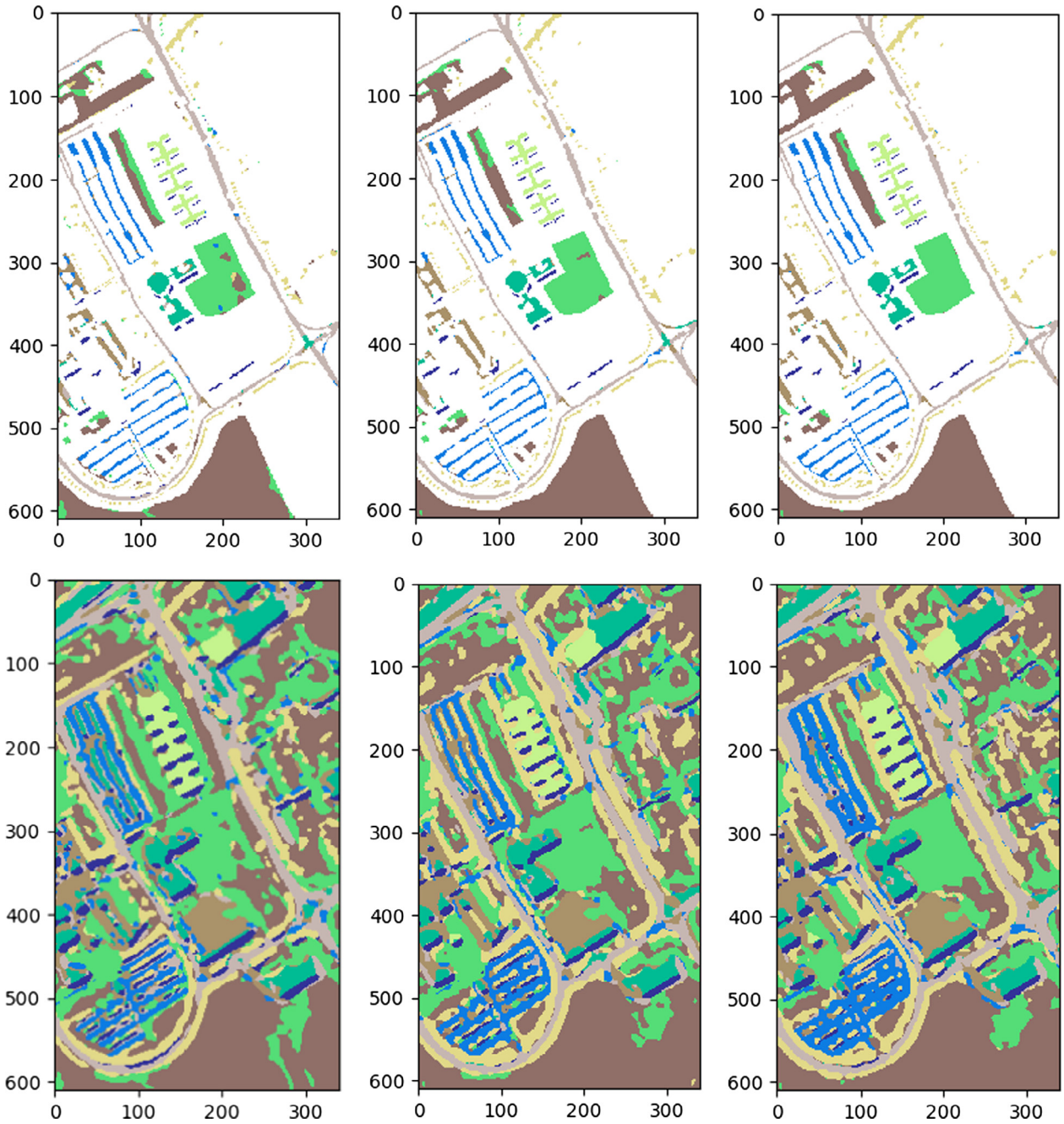


Fig. 18. Classification results for the University of Pavia data set with $d = 27$ and 50 (left), 100 (center) and 200 (right) samples per class. The upper row displays the classification result without the background, and the lower row displays the classification result with the background.

On the other hand, the results for the University of Pavia dataset are shown in Table 6. For each patch size, we can observe that the accuracy improves as the number of samples per class increases, reaching the best accuracy results with 200 samples per class. If we look for the most suitable patch size, we can say that $d = 21$ with 200 samples per class can be considered the best, with similar and balanced accuracy data. This is because, after adding more neighbors to the pixel, the data become more homogeneous.

In Figs. 16–18 we report the classification maps obtained for the University of Pavia dataset obtained in each experiment. The first one, Fig. 16 shows the Pavia classification results with a patch size of $d = 15$. Here, the accuracy becomes better when the number of samples per class is increased, although there are many mixed pixels in the three cases, especially when the background is added to the classification. We can observe a poor result on top of the meadows class, and the ones adjacent to the light green bare soil at the lowest part of the image.

Fig. 16 shows the Pavia classification results with a patch size of $d = 21$, which results in better results than $d = 15$. Pixels appear better defined, also with background. As expected, the best classification map is obtained with the maximum number of samples per class, i.e. 200, where we can see how the worst ranked pixels in the experiment with $d = 15$ are now better classified, e.g. the pixels at the top of the meadows class and the bare soil pixels.

Finally, in Fig. 18 the classification results with patch size $d = 27$ are shown. As in the previous cases, the result improves as more pixels are added to the training set (with 200 samples per class resulting in the best accuracy results). Although the pixels at the top of the image are better classified, the CNN finds it more difficult to classify the pixels of the meadow class in the center of the image, and the final result is slightly worse than the one obtained with patch size of $d = 21$.

4.5. Comparison with other algorithms

In this section we show comparisons of our proposed method with other existing methods, including a standard MLP and the 1-D, 2-D and 3-D CNNs in Chen et al. (2016). This represents an exhaustive and complete validation of our method with state-of-the-art CNNs in the hyperspectral imaging literature. Tables 7 and 8 respectively show the configurations (for both scenes) of the MLP and CNNs used in experiments for comparison purposes.

Comparison with MLP: For the MLP, the chosen topology is a *single layer feedforward network* (SLFN) with three layers: an input layer which receives a pixel in all its bands, a hidden layer whose number of nodes is calculated by $(n\_bands + n\_classes) \cdot \frac{2}{3}$, with a ReLU as activation function, and an output layer with the number of nodes equals as the number of classes and a *softmax* function. So, the final MLP topology for the Indian Pines hyperspectral data set is $200 - 144 - 16$ and for the University of Pavia data set is $103 - 75 - 9$, both with Adam optimizer (Kingma and Ba, 2014) and 0.0045 of learning rate. The proposed topologies have been

Table 7
Configuration of the MLP used in experiments for comparative purposes.

MLP Topologies			
Hyperspectral datasets	Layers		
	Type	N° neurons	Activation Function
Indian Pines	Input	200 *(n° bands)*	-
	Hidden	144 *(optimal n°)*	ReLU
	Output	16 *(n° classes)*	Softmax
University of Pavia	Input	103 *(n° bands)*	-
	Hidden	75 *(optimal n°)*	ReLU
	Output	9 *(n° classes)*	Softmax
Common parameters			
Batch size	Iterations	Learning rate	Optimizer
100	5000	0.045	AdamOptimizer

Table 8
Configuration of the CNNs used in experiments for comparative purposes. These are the 1-D, 2-D and 3-D CNNs configurations described in Chen et al. (2016).

1-D CNN Topologies (Chen et al., 2016)			
Hyperspectral datasets	Conv. Layers	ReLU	Pooling
Indian Pines	1×5	Yes	1×2
	1×5	Yes	1×2
	1×4	Yes	1×2
	1×5	Yes	1×2
	1×4	Yes	1×2
University of Pavia	1×8	Yes	1×2
	1×7	Yes	1×2
	1×8	Yes	1×2

2-D CNN Topologies (Chen et al., 2016)				
Hyperspectral datasets	Conv. Layers	ReLU	Pooling	Dropout
Indian Pines	$32 \times 4 \times 4$	Yes	2×2	No
	$64 \times 5 \times 5$	Yes	2×2	50%
	$128 \times 4 \times 4$	Yes	No	50%
University of Pavia	$32 \times 4 \times 4$	Yes	2×2	No
	$64 \times 5 \times 5$	Yes	2×2	50%
	$128 \times 4 \times 4$	Yes	No	50%

3-D CNN Topologies (Chen et al., 2016)				
Hyperspectral datasets	Conv. Layers	ReLU	Pooling	Dropout
Indian Pines	$128 \times 4 \times 4 \times 32$	Yes	2×2	No
	$192 \times 5 \times 5 \times 32$	Yes	2×2	50%
	$256 \times 4 \times 4 \times 32$	Yes	No	50%
University of Pavia	$32 \times 4 \times 4 \times 32$	Yes	2×2	No
	$64 \times 5 \times 5 \times 32$	Yes	2×2	50%
	$128 \times 4 \times 4 \times 32$	Yes	No	50%

Table 9
Execution times and accuracies obtained by the MLP (with configuration: 200 – 144 – 16) for the Indian Pines scene and by the MLP (with configuration: 103 – 75 – 9) for the University of Pavia scene, using 50, 100 and 200 samples per class. This experiment has been executed using 5000 iterations.

Datasets	Samples	Time per step		Total time		Accuracy	
		average	Std. deviation	average	Std. deviation	average	Std. deviation
Indian Pines	50	0.0036	0.0002	0.1791	0.0148	74.60	1.60
	100	0.0035	0.0001	0.1757	0.0149	79.29	1.35
	200	0.0036	0.0002	0.1800	0.0148	82.56	1.23
University of Pavia	50	0.0033	0.0001	0.1672	0.0147	82.79	1.92
	100	0.0031	0.0001	0.1550	0.0149	87.16	0.93
	200	0.0031	0.0001	0.1530	0.0149	87.76	1.75

tested with three different numbers of samples per class: 50, 100 and 200, repeating each one five times. Results are shown in Table 9, where we show the average time and accuracy of each execution with 5000 iterations and repeated five times. As we can see, as we add training data the accuracy increases, however the execution time remains fairly stable.

Table 10
Classification accuracies (and standard deviation) obtained by MLP with 50, 100 and 200 samples per class for the Indian Pines hyperspectral dataset. This experiment has been executed using 5000 iterations.

Neural network	MLP		
Samples per class	50	100	200
Alfalfa	98.26 (2.54)	98.70 (1.06)	97.39 (0.87)
Corn-notill	63.40 (4.60)	71.11 (3.74)	78.36 (4.46)
Corn-min	66.00 (3.58)	82.53 (2.47)	86.17 (3.31)
Corn	86.16 (2.77)	91.31 (2.08)	92.41 (2.28)
Grass/Pasture	90.52 (2.67)	91.88 (2.30)	96.31 (1.27)
Grass/Trees	93.45 (1.35)	95.64 (1.45)	97.73 (0.94)
Grass/pasture-mowed	97.14 (2.67)	97.86 (2.86)	97.86 (2.86)
Hay-windrowed	96.32 (1.47)	97.45 (0.80)	98.62 (0.54)
Oats	99.00 (2.00)	98.00 (4.00)	100.00 (0.00)
Soybeans-notill	75.12 (4.54)	83.87 (2.11)	87.00 (2.33)
Soybeans-min	62.81 (2.86)	62.42 (5.28)	68.99 (3.70)
Soybean-clean	79.39 (1.54)	84.69 (1.13)	87.86 (1.84)
Wheat	98.54 (0.31)	99.22 (0.59)	99.41 (0.37)
Woods	83.65 (4.78)	91.07 (1.55)	94.15 (2.29)
Bldg-Grass-Tree-Drives	74.46 (1.94)	82.75 (2.30)	85.03 (4.89)
Stone-steel towers	98.49 (1.10)	99.35 (0.53)	99.35 (0.53)
Overall Accuracy	75.24 (1.51)	80.34 (1.10)	84.60 (0.71)
Average Accuracy	85.17 (1.21)	89.24 (0.41)	91.66 (0.29)
Kappa	72.18 (1.51)	77.93 (1.10)	82.65 (0.71)
Runtime (sec.)	0.1791	0.1757	0.1800

In Table 10 we can observe in detail the results obtained for each class of the Indian Pines dataset. After 5000 iterations, we can see that the best overall accuracy result (84.60%) is obtained with the maximum number of samples per class, i.e. 200, without an overwhelming time difference. However, even with 5000 iterations, the MLP is still not able to reach 90% overall accuracy (although it reaches a 91.66% of average accuracy). If we pay attention to Fig. 19, we can see how the MLP needs fewer iterations to reach a low error as the number of samples per class increases in training, with a very little difference in execution times: each iteration of MLP with 200 samples per class is only 1.01 times slower than with 50 samples and 1.02 times slower than with 100 samples per class, as we can see in Fig. 20. Note how the optimizer in the first second quickly evolves from a very high initial error to a more reasonable value, given the cost function under which it is iterating.

Now we can compare the MLP classifier with our proposed CNN for the Indian Pines scene. We have considered two experiments: in the first one, we compared the MLP with 100 samples per class with the proposed CNN with also 100 samples per class and patch sizes of $d = 9, d = 19$ and $d = 29$. Each classifier has been executed five times with 1500 iterations. In Fig. 21 we can observe the evolution of the error in terms of the number of iterations of the MLP and the CNN. We can conclude that our CNN needs significantly less iterations to reach a low error when it uses patches of size $d = 29$ and $d = 19$ (specifically, the CNN reaches an error below

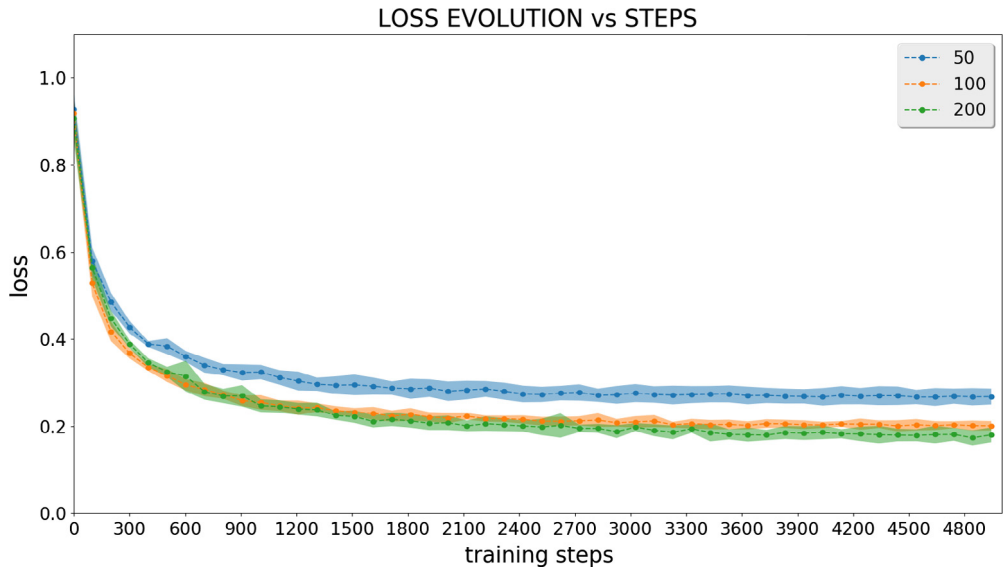


Fig. 19. Evolution of the validation error in terms of steps (iterations) with 50, 100 and 200 samples per class for the MLP with the Indian Pines scene. The shadow shows the standard deviation of the loss for the five executions of each patch (zoom in error interval $[0, 1]$). This experiment has been executed using 5000 iterations.

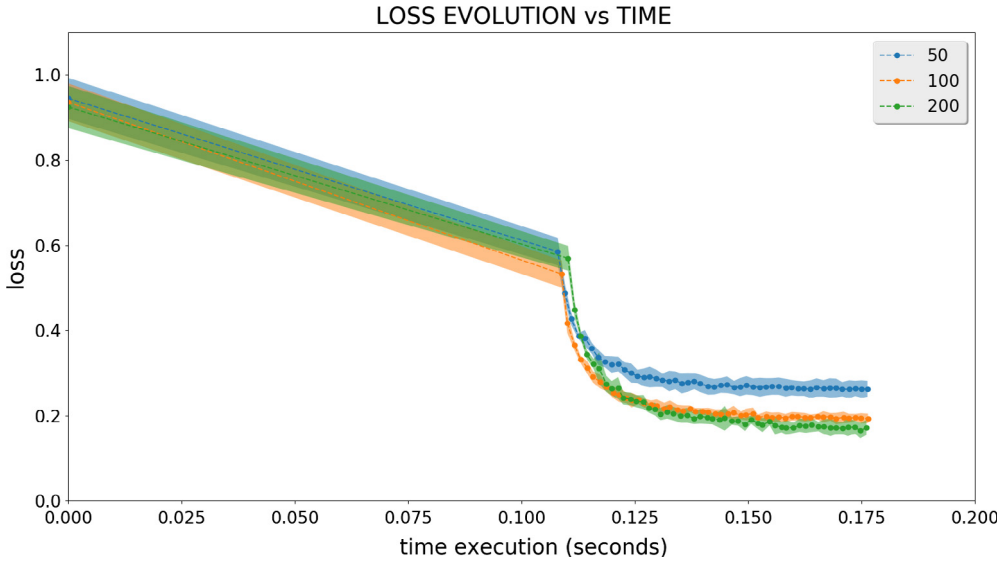


Fig. 20. Evolution of the validation error in terms of time (seconds) with 50, 100 and 200 samples per class for the MLP with the Indian Pines scene. The shadow shows the standard deviation of the loss for the five executions of each patch (zoom in error interval $[0, 1]$). This experiment has been executed using 5000 iterations.

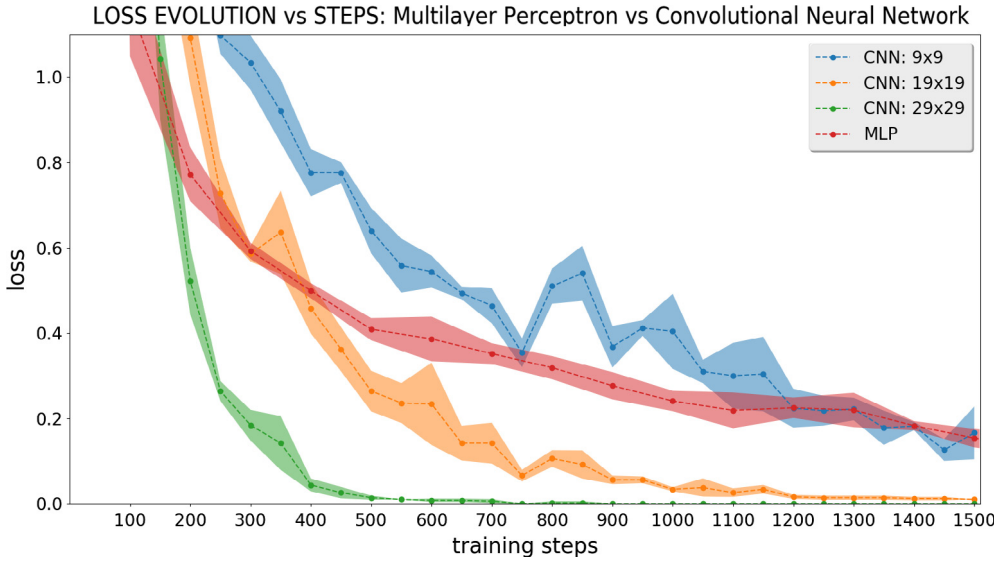


Fig. 21. Evolution of the validation error for the MLP and CNN (with $d = 9, d = 19$ and $d = 29$) in terms of steps for the Indian Pines image. The shadow shows the standard deviation of the loss for each network repeated five times. Zoom in $(0, 1)$. This experiment has been executed using 1500 iterations.

0.1 with only 300 iterations when $d = 29$ and around 700–800 iterations when $d = 19$) while the MLP barely drops from 0.2 in 1500 iterations. However, in Fig. 22 (in which we show the error evolution in terms of time in seconds) we can observe that one iteration of the CNN (with any patch size) is always slower than one iteration of the MLP.

The second comparison between the MLP and our CNN is reported in Table 12. In this case we used 200 samples per class for the MLP and CNN, and patches of size $d = 9, d = 19$ and $d = 29$. Table 12 shows that the MLP is the fastest classification method (all its executions take around 0.17–0.18 s), reaching its better average and overall accuracy values (91.66% and 84.60%, respectively) with 200 samples per class. However, these results are several points lower than the accuracies reached by the proposed CNN with patches of size $d = 19$ and $d = 29$. Specifically, the MLP reaches an overall accuracy around 14 points lower than the CNN, and an average accuracy around 8 points lower than the one achieved by the CNN for $d = 29$. With $d = 19$, the MLP reaches an overall accuracy around 11 lower than that achieved by the CNN and an average accuracy around 7 points lower than that achieved by the CNN. Only if we compare the MLP with the CNN and $d = 9$ the MLP reaches better overall and average accuracies: around 3 points better than the overall accuracy achieved by the CNN and around 1.5 points better than the average accuracy achieved by the CNN.

The resulting classification maps obtained by the MLP are shown in Fig. 23, where the classification results without background (top row) and with background (lower row) are reported. In both cases, unlike the CNN, the MLP results are not well defined, with many pixels of different classes appearing mixed in the final classification. An increase in the number of samples per class slightly improves the classification results, without reaching the quality of the classification maps provided by the CNN in Figs. 13–15.

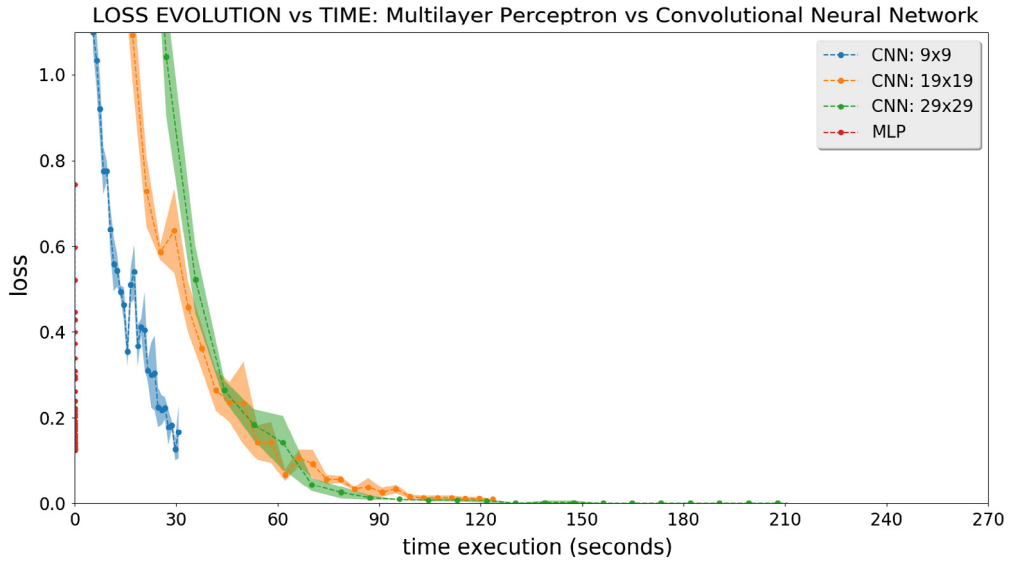


Fig. 22. Evolution of the validation error for the MLP and CNN (with $d = 9, d = 19$ and $d = 29$) in terms of time (seconds) for the Indian Pines image. The shadow shows the standard deviation of the loss for each network repeated five times. Zoom in $(0, 1)$. This experiment has been executed using 1500 iterations.

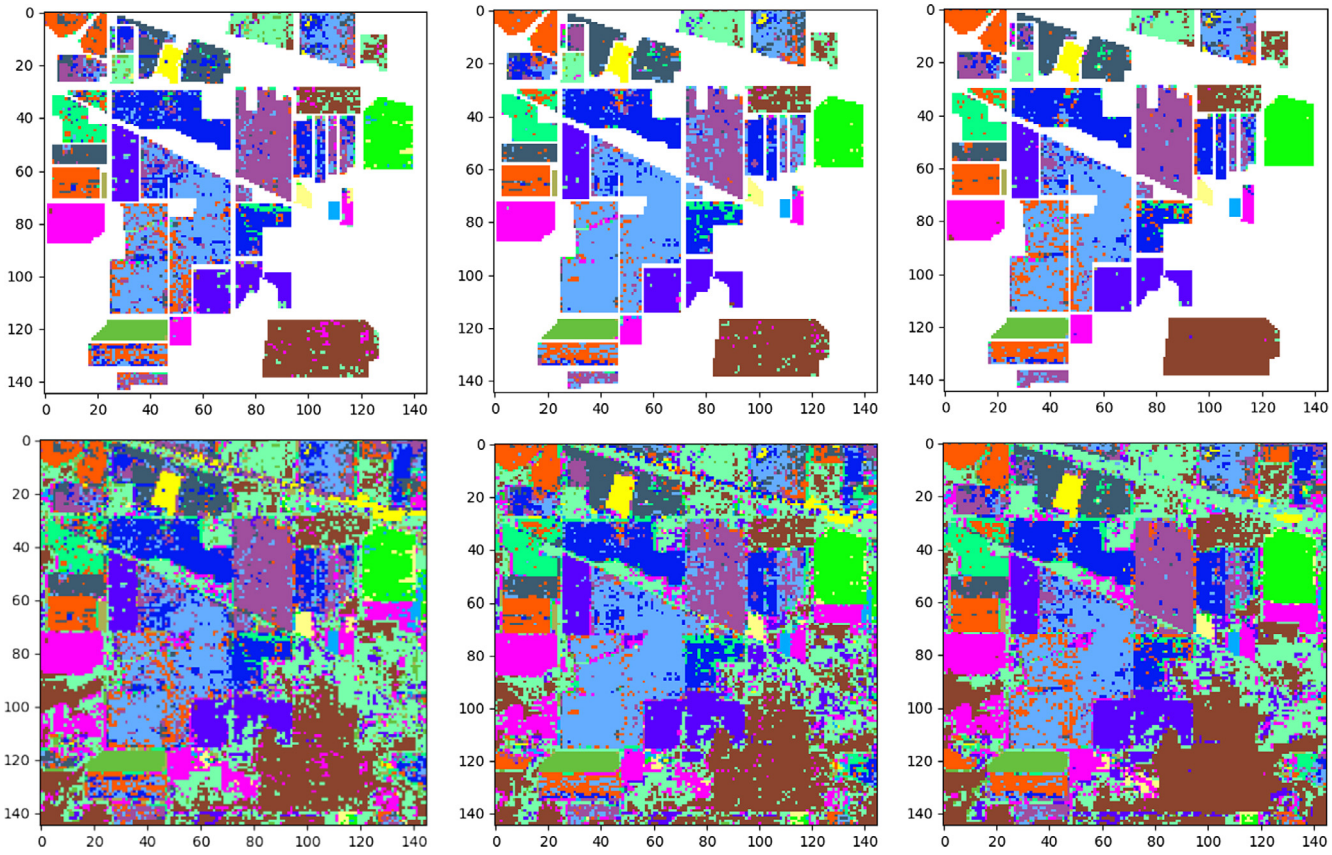


Fig. 23. Indian Pines classification results achieved by the MLP with 50 (left), 100 (center) and 200 (right) samples per class with 1500 iterations. The upper row displays the classification result without the background, and the lower row displays the classification result with the background.

Also, Table 9 summarizes the experiments conducted using the MLP classifier with the University of Pavia dataset for 50, 100 and 200 samples per class (all repeated five times, with 5000 iterations per execution). We can observe that the best accuracy result (87.76%) is obtained with the maximum number of samples per class, which is 200, as it was already the case with the Indian Pines image. The execution times for 50, 100 and 200 samples per class are very similar too. However, with 5000 iterations the MLP is also unable to reach 90% accuracy. In Fig. 24 we can see how the error descends as the MLP iterates, needing less iterations as the number of samples in the training increases, with a very little difference in execution times as we can see in Fig. 25. Also, in Table 11 we can

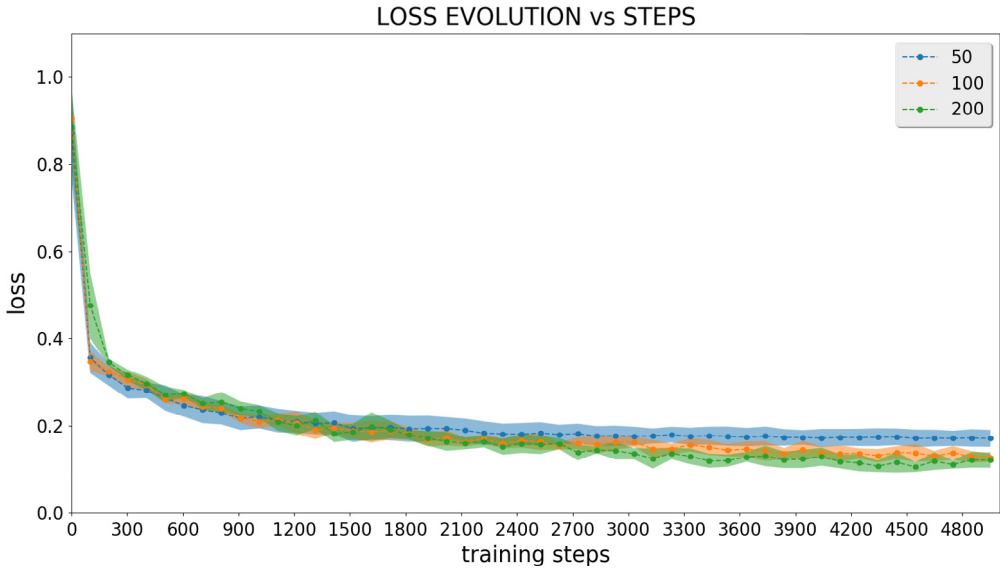


Fig. 24. Error evolution for the MLP in terms of steps (iterations) with 50, 100 and 200 samples per class for the University of Pavia data set. The shadow shows the standard deviation of the loss for the five executions of each patch (zoom in error interval $[0, 1]$).

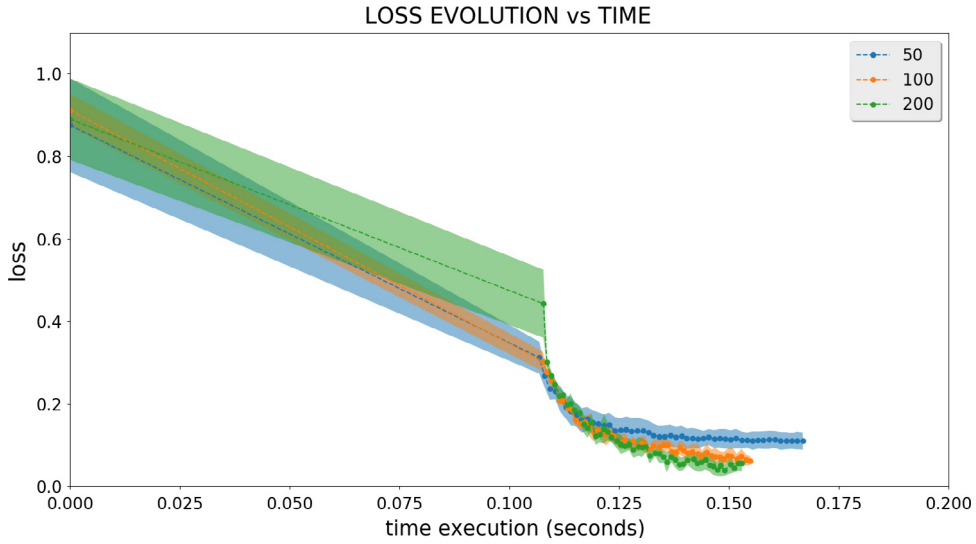


Fig. 25. Error evolution for the MLP in terms of time (seconds) with 50, 100 and 200 samples per class for the University of Pavia data set. The shadow shows the standard deviation of the loss for the five executions of each patch (zoom in error interval $[0, 1]$).

Table 11
Classification accuracies obtained by MLP with 50, 100 and 200 samples per class for the University of Pavia dataset.

Neural network	MLP		
Samples per class	50	100	200
Asphalt	81.29 (1.15)	83.34 (1.18)	84.92 (0.87)
Meadows	82.83 (4.46)	85.44 (2.06)	89.56 (3.75)
Gravel	84.71 (1.79)	85.98 (4.89)	94.68 (1.41)
Trees	91.64 (2.03)	94.77 (0.50)	96.48 (1.54)
Painted metal sheets	99.18 (0.16)	99.38 (0.36)	99.43 (0.30)
Bare Soil	84.87 (3.31)	88.94 (2.89)	90.75 (2.85)
Bitumen	89.35 (3.10)	92.63 (0.18)	93.76 (0.90)
Self-Blocking Bricks	79.57 (2.35)	79.50 (6.26)	63.94 (3.50)
Shadows	99.79 (0.09)	99.54 (0.36)	99.96 (0.05)
Overall Accuracy	84.37 (2.30)	86.68 (0.67)	88.20 (1.50)
Average Accuracy	88.14 (1.05)	89.95 (0.38)	90.39 (0.30)
Kappa	79.87 (2.30)	82.79 (0.67)	84.67 (1.50)
Runtime (sec.)	0.1672	0.1550	0.1530

observe the accuracy results for each class obtained after executing five times the MLP with the Pavia dataset.

Now, we can compare the results obtained by the MLP over the University of Pavia data set with the results obtained by the considered CNN architectures. In order to do so, we have performed two experiments: in the first one we execute the MLP with 100 samples per class and we choose a patch size of $d = 15, d = 21$ and $d = 27$ for the CNN, and further execute it with 100 samples per class too. Each experiment has been run using 1500 iterations. In Fig. 26 we can see that the MLP needs more iterations than the CNN to reduce its associated error, without reaching in any case the error achieved by the CNN, although its iterations are faster than those of the CNN, as we can observe in Fig. 27, where it is shown that the execution tine of the MLP takes less than one second.

The second experiment reports a comparison between the MLP and the proposed CNN with 200 samples per class for both

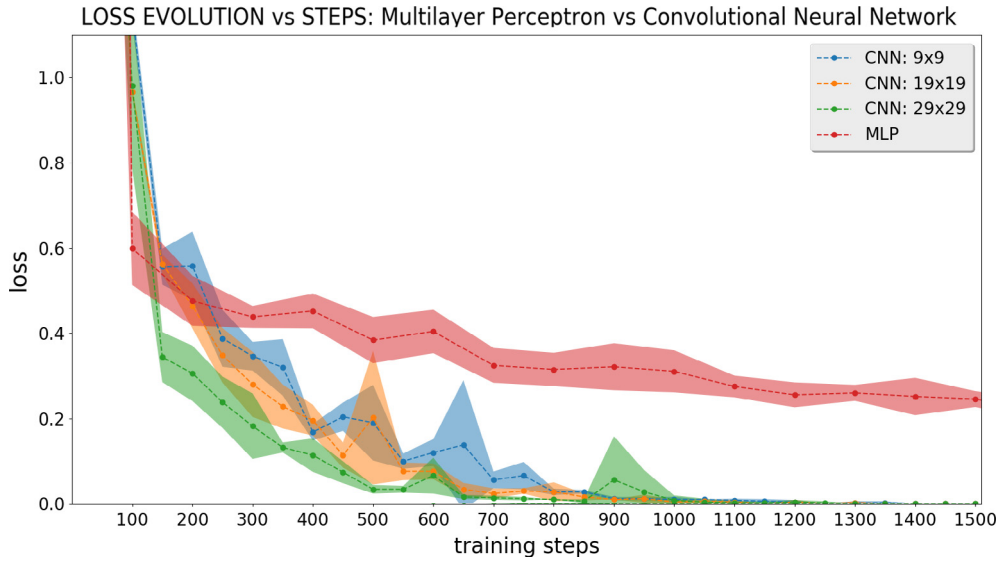


Fig. 26. Error evolution for the MLP and CNN (with $d = 15, d = 21$ and $d = 27$) in terms of steps for the University of Pavia data set. The shadow shows the standard deviation of the loss for each network repeated five times. Zoom in $(0, 1)$.

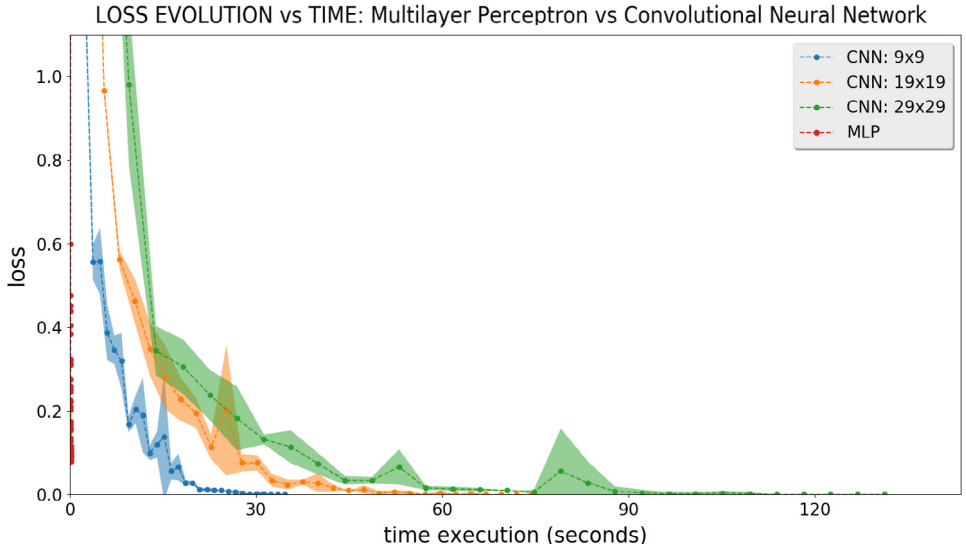


Fig. 27. Error evolution for the MLP and CNN (with $d = 15, d = 21$ and $d = 27$) in terms of time (seconds) for the University of Pavia data set. The shadow shows the standard deviation of the loss for each network repeated five times. Zoom in $(0, 1)$.

classifiers, and patch sizes of $d = 15, d = 21$ and $d = 27$ for the CNN. Table 13 shows that the MLP is the fastest classification method, even faster than the MLP with Indian Pines (all its executions take around 0.15–0.16 s), reaching its best average ad overall values (91% and 89% respectively) with 200 samples per class. But, again, the CNN reaches better accuracy values, with an average accuracy of 98% and overall accuracy of 97%, i.e. around 8–9% points better than MLP due to the inability of the latter architecture to improve its outcome.

In Fig. 28 we can observe the MLP classification maps for the University of Pavia. The top images show the classification with ground-truth pixels, whose result improves as more samples are added in the training. However the classification maps are very mixed as compared to those obtained by the CNN in Figs. 16–18. On the other hand, the bottom images show the classification with the whole background. In this case, the areas of the image are relatively well distinguished (even better when more samples per class are added), although the number of mixed pixels is greater than in the CNN experiments with $d = 27, d = 21$ and $d = 15$.

Comparison with other convolutional networks: Now we compare our proposed CNN with other deep architectures, in particular with the 1-D, 2-D and 3-D CNNs described in Chen et al. (2016). In this work the authors studied the application of supervised CNNs in hyperspectral imaging feature extraction. Three deep feature extraction architectures based on the CNN were proposed to extract the spectral, spatial, and spectral-spatial features of hyperspectral imaging, respectively. To address the overfitting problem caused by the limited number of training samples, the authors implemented some regularization strategies, including L2 regularization and dropout in the training process. Also, they proposed a virtual sample enhanced method to create training samples. The main differences between our method and the one described in Chen et al. (2016) can be summarized in the following points:

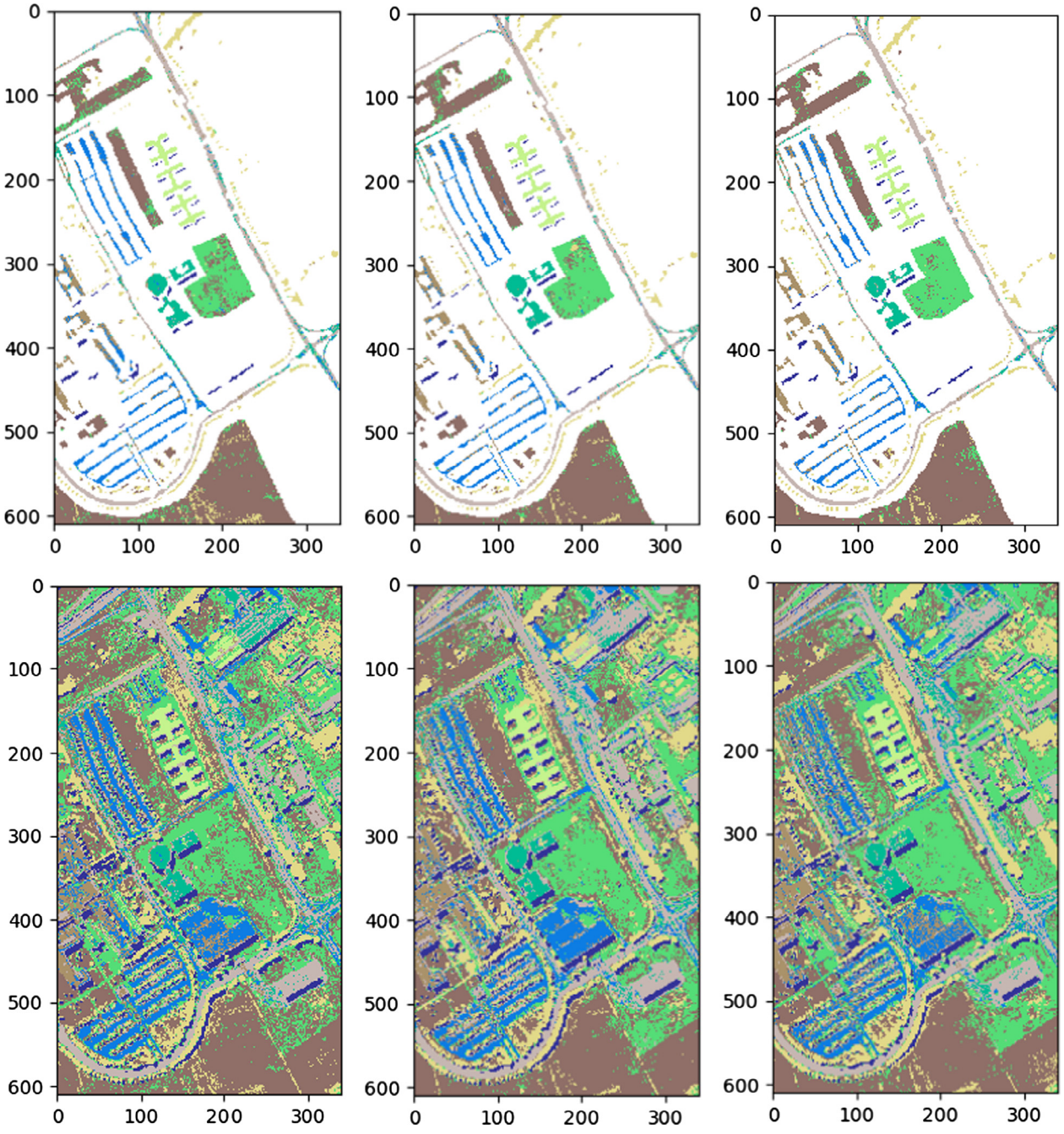


Fig. 28. Classification results achieved by the MLP for the University of Pavia data set with 50, 100 and 200 samples per class with 1500 iterations. The upper row displays the classification result without the background, and the lower row displays the classification result with the background.

- Regarding the configuration of the 1-D CNN, we provide a detailed description in Table 8. For the Indian Pines dataset, the learning rate of the 1-D CNN is fixed to 0.005 with 700 training epochs, while for University of Pavia dataset, the learning rate is fixed to 0.001 with 600 epochs. The datasets are divided into training and testing sets. For the Indian Pines dataset, 1765 labeled pixels are chosen to create the training set, while for the University of Pavia dataset, the authors use 3930. This spectral-CNN receives a normalized pixel vector (1×200 if it is an Indian Pines pixel and 1×103 if it is an University of Pavia pixel) in the range $[-1, 1]$. The data suffers a L2 regularization along the CNN and, at the end of the CNN procedure, the input pixel vector is converted into a feature vector that is fed to *Logistic Regression* (LR) for classification. The authors selected a mini-batch update strategy, and the cost function is calculated on a mini-batch of inputs as $c_o = -\frac{1}{m}\sum_{i=1}^{m}[x_i \log(z_i) + (1 - x_i)\log(1 - z_i)]$, using mini-batch stochastic gradient descent as optimizer of the 1-D CNN.
- Regarding the configuration of the 2-D CNN, we provide a detailed description in Table 8. This spatial-CNN receives, through a preprocessing with PCA, patches of size 27×27 normalized in the range $[-0.5, 0.5]$ and grouped in batches of 100. The output of the CNN is a feature vector of 1×128 that is sent to LR for classification. As in the previous 1-D CNN, the input image is represented by some feature vectors, which capture the spatial information contained in the neighborhood region of the input pixel. Then, the learned features are fed to the LR for classification.

Table 12
Classification accuracies obtained by different neural networks tested using the Indian Pines dataset: (1) first column: results obtained by the MLP (trained with 200 samples per class); (2) second column: comparison between the results obtained by the 1-D CNN, 2-D CNN and 3-D CNN in Chen et al. (2016) and the results obtained by our CNN (trained with the same number of samples per class as the CNNs in Chen et al. (2016)), using different values of parameter d; (3) third column: results obtained by our CNN (trained with 200 samples per class, using different values of parameter d).

Accuracy table													
Neural networks	MLP		CNNs in Chen et al., 2016 versus the proposed CNN							Proposed CNN			
	SLFN	Samples	1-D	2-D	3-D	$d=9$	$d=19$	$d=29$	Samples	$d=9$	$d=19$	$d=29$	Samples
Alfalfa	97.39	33	89.58	99.65	100.00	100.00	100.00	100.00	30	99.13	99.57	99.13	33
Corn-notill	78.36	200	85.68	90.64	96.34	90.57	94.06	97.17	150	80.48	94.47	98.17	200
Corn-min	86.17	200	87.36	99.11	99.49	97.69	96.43	98.17	150	96.65	98.22	98.92	200
Corn	92.41	181	93.33	100.0	100.00	99.92	100.00	100.00	100	99.66	100.00	100.00	181
Grass/Pasture	96.31	200	96.88	98.48	99.91	98.10	98.72	98.76	150	99.46	99.75	99.71	200
Grass/Trees	97.73	200	98.99	97.95	99.75	99.34	99.67	100.00	150	99.53	98.90	99.40	200
Grass/pasture-mowed	97.86	20	91.67	100.00	100.00	100.00	100.00	100.00	20	100.00	100.00	100.00	20
Hay-windrowed	98.62	200	99.49	100.00	100.00	99.58	99.92	100.00	150	99.67	99.62	100.00	200
Oats	100.00	14	100.00	100.00	100.00	100.00	100.00	100.00	15	100.00	100.00	100.00	14
Soybeans-notill	87.00	200	90.35	95.33	98.72	94.28	97.63	99.14	150	92.43	98.00	98.62	200
Soybeans-min	68.99	200	77.90	78.21	95.52	87.75	92.93	94.59	150	76.42	94.32	96.15	200
Soybean-clean	87.86	200	95.82	99.39	99.47	94.81	97.17	99.06	150	97.74	99.09	99.33	200
Wheat	99.41	143	98.59	100.00	100.00	100.00	100.00	100.00	150	99.71	100.00	99.90	143
Woods	94.15	200	98.55	97.71	99.55	98.09	97.88	99.76	150	97.71	98.85	98.96	200
Bldg-Grass-Tree-Drives	85.03	200	87.41	99.31	99.54	89.79	95.80	98.39	50	99.27	99.90	100.00	200
Stone-steel towers	99.35	75	98.06	99.22	99.34	100.00	99.57	98.92	50	100.00	100.00	100.00	75
Overall Accuracy (OA)	84.60		87.81	89.99	97.56	93.94	96.29	97.87		90.11	97.23	98.37	
Average Accuracy (AA)	91.66		93.12	97.19	99.23	96.87	98.11	99.00		96.12	98.79	99.27	
Kappa	82.65		85.30	87.95	97.02	93.12	95.78	97.57		88.81	96.85	98.15	
Runtime (sec.)	0.1800		457.8	357.0	1675.2	74.47	189.51	158.42		48.21	197.65	405.46	
Total samples		2466							1765				2466

- Finally, the configuration of the 3-D CNN is shown in Table 8. This spatial-spectral CNN receives patches of size $27 \times 27 \times n_{bands}$ normalized in range $[-0.5, 0.5]$ and grouped in batches of 100. In this case, n_{bands} is fixed to 32. The learning rate is fixed to 0.003 and the training epochs is set to 400. After convolutional and pooling layers, the input data is transformed and fed to LR for classification.

In Table 12 we can see a detailed comparison between the different tested neural networks using Indian Pines dataset. The first column reports the results obtained by the MLP (trained with 200 samples per class). The second column provides a comparison between the CNNs in Chen et al. (2016) and our CNN (using the same number of samples per class as the methods in Chen et al. (2016)), with patch sizes of $d = 9, d = 19$ and $d = 29$. Finally, in the third column we also report the results obtained by our proposed CNN (trained with 200 samples per class) for comparison.

If we focus on analyzing the results provided by the different implementations of Chen et al. (2016) in the second column, we

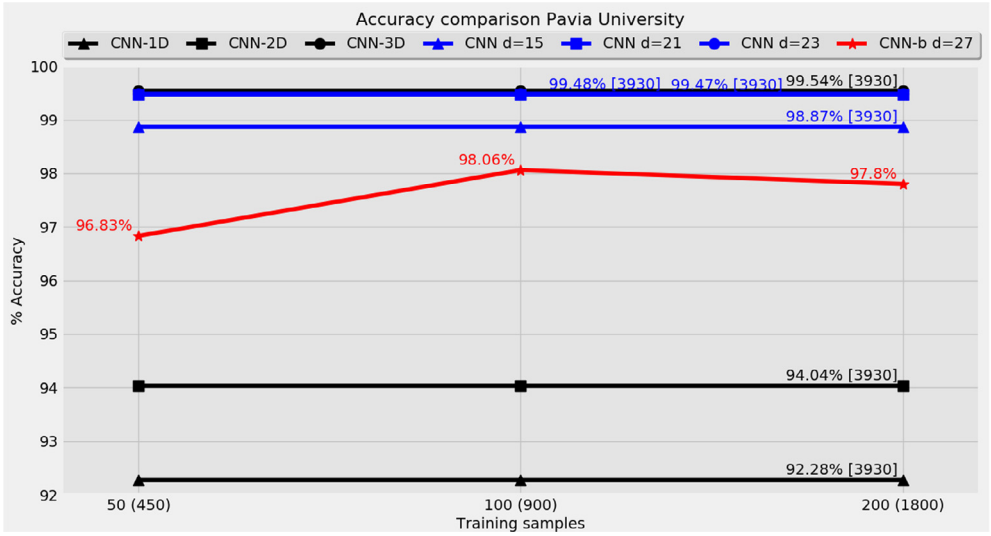


Fig. 29. Comparison of the overall accuracy achieved by CNN classifiers with the Indian Pines scene. The horizontal black lines show the overall accuracy results reached by the 1-D, 2-D and 3-D CNNs in Chen et al. (2016), and the horizontal blue lines show the overall accuracy results obtained by our proposed CNN, implemented with different values of d and trained with the same number of samples than the CNNs in Chen et al. (2016). The red line (marked as CNN-b in the figure) corresponds to our CNN, implemented with $d = 29$ but trained with 50, 100 and 200 samples per class. Above each black and blue line, we report the number of used samples and the overall accuracy reached (in square brackets). For the red line, we only report the overall accuracy reached (the number of used samples for this line is defined by the x-axis). On the other hand, the y-axis shows the overall accuracies obtained in the experiments. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

can see that the one with spatial information (2-D CNN) achieves better results than the one with spectral information (1-D CNN). Also, we can see that the inclusion of the two sources of information (3-D CNN) leads to an overall improvement of the accuracy. In the same column we can observe that, when using our CNN (with the same number of samples per class as the methods in Chen et al. (2016)), the increase in the value of parameter d leads to an improvement in the obtained classification result. Also, our method is faster than all the methods reported in Chen et al. (2016) and comparable in terms of overall accuracy to the best methods reported in that work. Specifically, our proposed CNN implemented with $d = 29$ is 2.89 times faster than the 1-D CNN, with an overall accuracy that is 10.06 percentage points better; 2.25 times faster than the 2-D CNN, with an overall accuracy 7.88 percentage points better; and 10.57 times faster than the 3-D CNN, with very similar overall accuracy. Finally, a comparison between the results in the first and third columns of Table 12 indicate that our proposed CNN can achieve better results in terms of overall accuracy than the MLP, but the MLP is faster.

Also in Fig. 29 we provide a graphical comparison of the overall accuracy results obtained by the proposed CNN the CNNs implemented by Chen et al. (2016). The horizontal blue lines in Fig. 29 show the overall accuracy results obtained by our proposed CNN with different values of d and trained using the same number of samples than the CNNs in Chen et al. (2016). The red line corresponds to our CNN, implemented with $d = 29$ but trained with 50, 100 and 200 samples per class. As we can see, the proposed CNN can reach better overall accuracies than the compared 1-D and 2-D CNNs. For the 3-D CNN, the results can be comparable in terms of overall accuracy. However, since our architecture is optimized and executed on a GPU, we can get better results from the viewpoint of processing time.

Table 13
Classification accuracies obtained by different neural networks tested using the University of Pavia dataset: (1) first column: results obtained by the MLP (trained with 200 samples per class); (2) second column: comparison between the results obtained by the 1-D CNN, 2-D CNN and 3-D CNN in Chen et al. (2016) and the results obtained by our CNN (trained with the same number of samples per class as the CNNs in Chen et al. (2016)), using different values of parameter d; (3) third column: results obtained by our CNN (trained with 200 samples per class, using different values of parameter d).

Accuracy table													
Neural networks	MLP		CNNs in Chen et al., 2016 versus the proposed CNN							Proposed CNN			
	SLFN	Samples	1-D	2-D	3-D	$d = 15$	$d = 21$	$d = 23$	Samples	$d = 15$	$d = 21$	$d = 27$	Samples
Asphalt	84.92	200	92.06	97.11	99.36	97.53	98.80	98.59	548	92.81	95.31	96.31	200
Meadows	89.56	200	92.80	87.66	99.36	98.98	99.46	99.60	540	97.20	98.16	97.54	200
Gravel	94.68	200	83.67	99.69	99.69	98.96	99.59	99.45	392	96.97	97.92	96.84	200
Trees	96.48	200	93.85	98.49	99.63	99.75	99.68	99.57	542	98.62	98.74	97.58	200
Painted metal sheets	99.43	200	98.91	100.00	99.95	99.93	99.78	99.61	256	100.00	100.00	99.65	200
Bare Soil	90.75	200	94.17	98.00	99.96	99.42	99.93	99.84	532	98.57	99.57	99.33	200
Bitumen	93.76	200	92.68	99.89	100.00	98.71	99.88	100.00	375	97.27	99.75	98.90	200
Self-Blocking Bricks	63.94	200	89.09	99.70	99.65	98.58	99.53	99.67	514	96.17	98.20	98.89	200
Shadows	99.96	200	97.84	97.11	99.38	99.87	99.79	99.83	231	99.86	99.82	99.58	200
Overall Accuracy (OA)	88.20		92.28	94.04	99.54	98.87	99.47	99.48		96.83	98.06	97.80	
Average Accuracy (AA)	90.39		92.55	97.52	99.66	99.08	99.60	99.57		97.50	98.61	98.29	
Kappa	84.67		90.37	92.43	99.41	98.51	99.30	99.32		95.83	97.44	97.09	
Runtime (sec.)	0.15		994.80	607.19	2769.00	43.16	94.57	107.56		42.83	74.09	132.68	
Total samples		1800							3930				1800

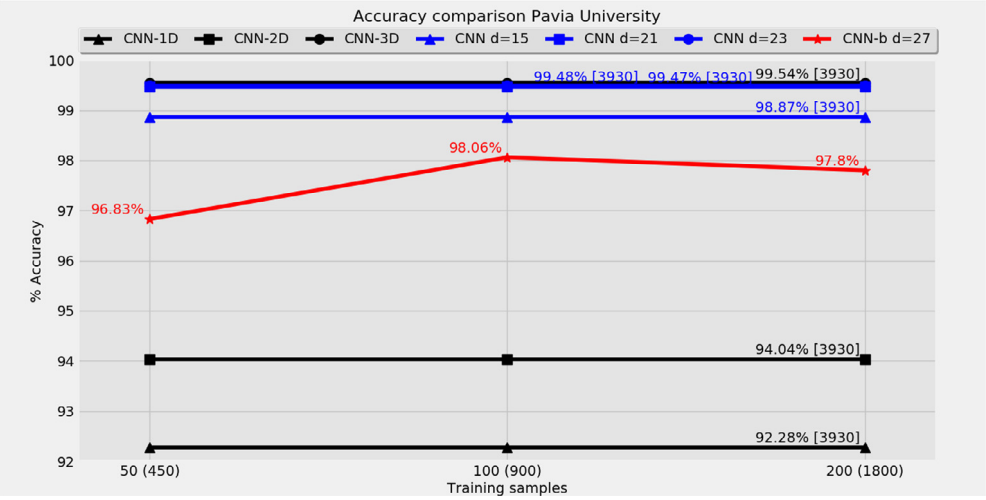


Fig. 30. Comparison of the overall accuracy achieved by CNN classifiers with the University of Pavia scene. The horizontal black lines show the overall accuracy results reached by the 1-D, 2-D and 3-D CNNs in Chen et al. (2016), and the horizontal blue lines show the overall accuracy results obtained by our proposed CNN, implemented with different values of d and trained with the same number of samples than the CNNs in Chen et al. (2016). The red line (marked as CNN-b in the figure) corresponds to our CNN, implemented with $d = 29$ but trained with 50, 100 and 200 samples per class. Above each black and blue line, we report the number of used samples and the overall accuracy reached (in square brackets). For the red line, we only report the overall accuracy reached (the number of used samples for this line is defined by the x-axis). On the other hand, the y-axis shows the overall accuracies obtained in the experiments. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

In Table 13 we report a comparison between the different tested neural networks using Pavia University data set. The first column reports the results obtained by the MLP (trained with 200 samples per class). The second column provides a comparison between the CNNs in Chen et al. (2016) and our CNN (using the same number of samples per class as the methods in Chen et al. (2016)), with patch sizes of $d = 15, d = 21$ and $d = 23$ (in this case, due to the number of samples per class used by Chen et al. (2016) there is not enough memory to run the CNN with $d = 27$). Finally, in the third column we also report the results obtained by our proposed CNN (trained with 200 samples per class and with patch sizes of $d = 15, d = 21$ and $d = 23$) for comparison.

Again, we can see in the second column that our method is faster than all the methods reported in Chen et al. (2016) and comparable in terms of overall accuracy to the best methods reported in that work. Specifically, our proposed CNN implemented with $d = 23$ is 9.25 times faster than the 1-D CNN with an overall accuracy that is 7.20 percentage points better; 5.65 times faster than the 2-D CNN with an overall accuracy that is 5.44 percentage points better; and 25.74 times faster than the 3-D CNN, with very similar overall accuracy. A comparison between the results in the first and third columns of Table 13 again reveals that our proposed CNN can achieve better results in terms of overall accuracy than the MLP, but the MLP is faster.

In Fig. 30 we can graphically compare the overall accuracies obtained by our proposed CNN with those obtained by the 1-D, 2-D and 3-D CNNs reported in Chen et al. (2016). Again, the horizontal blue lines in Fig. 30 show the overall accuracy results obtained by our proposed CNN, implemented with different values of d and trained using the same number of samples than the methods in Chen et al. (2016). The red line corresponds to our CNN implemented with $d = 27$ but trained with 50, 100 and 200 samples per class. As we can see, the proposed CNN can reach overall accuracies that are better than those achieved by the 1-D and 2-D CNNs, and comparable to those achieved by the 3-D CNN in Chen et al. (2016). However, the runtime of our implementation is considerably smaller, thanks to our GPU implementation of the network.

5. Conclusions and future work

In this paper, we have developed a new deep 3-D CNN architecture for spatial-spectral classification of hyperspectral data. The joint consideration of spectral information together with spatial information provides better classification results than those reached by traditional neural networks that only include spectral information. With a proper topology selection and a good election of parameters, we can obtain high classification accuracies in acceptable processing times, enforced by the fact that our CNN has been implemented efficiently using GPUs. Our detailed comparison with other 1-D, 2-D and 3-D CNNs in Chen et al. (2016) (that also include spatial and spectral information simultaneously) reveals a good compromise between the classification results obtained by our newly proposed CNN architecture and the time needed to obtain these results in the considered computing environments, which is important for practical exploitation of the proposed methodology in real applications. Our experiments specifically suggest that, with a proper and simple adaptation, the use of GPUs allows us to realize the full potential of deep learning techniques for remotely sensed hyperspectral image classification by naturally and efficiently combining the spatial and the spectral information contained in these images. This has also been verified with a classic MLP model used for comparative purposes in this work. As future work, we will conduct additional experiments with other hyperspectral scenes and also test other high performance computing architectures for efficient implementation.

Acknowledgement

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). Last but not least, the authors would like to take this opportunity to gratefully thank the Editors and the Anonymous Reviewers for their careful assessment of our manuscript and for their outstanding comments and suggestions, which greatly helped us to improve the technical quality and presentation of our work.

References

Atkinson, P.M., Tatnall, A.R.L., 1997. Introduction neural networks in remote sensing. Int. J. Remote Sens. 18 (4), 699–709. https://doi.org/10.1080/014311697218700.

Benediktsson, J.A., Swain, P.H., 1990. Statistical Methods and Neural Network Approaches for Classification of Data from Multiple Sources (Ph.D. thesis). Purdue Univ., School of Elect. Eng. West Lafayette, IN.

Benediktsson, J.A., Swain, P.H., Ersoy, O.K., 1993. Conjugate gradient neural networks in classification of very high dimensional remote sensing data. Int. J. Remote Sens. 14 (15), 2883–2903.

Bengio, Y., 2009. Learning deep architectures for AI. Mach. Learn. 2 (1), 1–127.

Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H., 2007. Greedy layer-wise training of deep networks. In: Schölkopf, B., Platt, J., Hoffman, T. (Eds.), Advances in Neural Information Processing Systems 19 (NIPS'06). MIT Press, pp. 153–160.

Bishop, C.M., 1995. Neural Networks for Pattern Recognition. Clarendon Press <https://books.google.es/books?id=-aAwQO_-rXwC>.

Böhning, D., 1992. Multinomial logistic regression algorithm. Ann. Inst. Stat. Math. 44 (1), 197–200.

Camps-Valls, G., Bruzzone, L., 2005. Kernel-based methods for hyperspectral image classification. IEEE Trans. Geosci. Remote Sens. 43 (6), 1351–1362.

Chang, C.-I., 2003. Hyperspectral Imaging: Techniques for Spectral Detection and Classification. Springer, US.

Chen, Y., Jiang, H., Li, C., Jia, X., Ghamisi, P., 2016. Deep feature extraction and classification of hyperspectral images based on convolutional neural networks. IEEE Trans. Geosci. Remote Sens. 54 (10), 6232–6251 <http://ieeexplore.ieee.org/document/7514991/>.

Chen, Y., Lin, Z., Zhao, X., Wang, G., Gu, Y., 2014. Deep learning-based classification of hyperspectral data. IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens. 7 (6), 2094–2107.

Cheng, G., Han, J., Lu, X., 2017. Remote sensing image scene classification: benchmark and state of the art. Proc. IEEE (99), 1–19.

Chien, Y., 1974. Pattern classification and scene analysis. IEEE Trans. Autom. Control 19 (4), 462–463.

Cho, K., 2014. Foundations and Advances in Deep Learning (Ph.D. thesis). Aalto University.

Deng, L., Yu, D., 2014. Deep learning: methods and applications. Found. Trends® Signal Process. 7 (3–4), 197–387.

Dosovitskiy, A., Springenberg, J.T., Riedmiller, M., Brox, T., 2014. Discriminative unsupervised feature learning with convolutional neural networks. In: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q. (Eds.), Advances in Neural Information Processing Systems, vol. 27. Curran Associates, Inc., pp. 766–774.

Duchi, J., Edu, J.B., Hazan, E., Singer, Y., 2011. Adaptive subgradient methods for online learning and stochastic optimization∗. J. Mach. Learn. Res. 12, 2121–2159.

Fauvel, M., Benediktsson, J.A., Chanussot, J., Sveinsson, J.R., 2008. Spectral and spatial classification of hyperspectral data using SVMs and morphological profiles. IEEE Trans. Geosci. Remote Sens. 46 (11), 3804–3814.

Fisher, P., 1997. The pixel: a snare and a delusion. Int. J. Remote Sens. 18 (3), 679–685.

García, V., Sánchez, J.S., Mollineda, R.A., 2011. Classification of high dimensional and imbalanced hyperspectral imagery data. In: Proceedings Pattern Recognition

and Image Analysis: 5th Iberian Conference, IbPRIA 2011, Las Palmas de Gran Canaria, Spain, June 8–10, 2011. Springer, Berlin, Heidelberg, pp. 644–651.
Ghamisi, P., Plaza, J., Chen, Y., Li, J., Plaza, A., 2017. Advanced supervised spectral classifiers for hyperspectral images: a review. IEEE Geosci. Remote Sens. Mag. 5 (1), 8–32.
Glorot, X., Bengio, Y., 2010. Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS10). Society for Artificial Intelligence and Statistics. pp. 249–256.
Glorot, X., Bordes, A., Bengio, Y., 2011. Deep sparse rectifier neural networks. In: Gordon, Geoffrey J., Dunson, David B. (Ed.), Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS-11). Journal of Machine Learning Research – Workshop and Conference Proceedings, pp. 315–323.
Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., 2014. Generative adversarial nets. In: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q. (Eds.), Advances in Neural Information Processing Systems, vol. 27. Curran Associates, Inc., pp. 2672–2680.
Green, R.O., Eastwood, M.L., Sarture, C.M., Chrien, T.G., Aronsson, M., Chippendale, B. J., Faust, J.A., Pavri, B.E., Chovit, C.J., Solis, M., Olah, M.R., Williams, O., 1998. Imaging spectroscopy and the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS). Remote Sens. Environ. 65 (3), 227–248.
Haut, J.M., Paoletti, M., Plaza, J., Plaza, A., 2017a. Cloud implementation of the k-means algorithm for hyperspectral image analysis. J. Supercomput. 73 (1), 514–529. https://doi.org/10.1007/s11227-016-1896-3.
Haut, J.M., Paoletti, M.E., Paz-Gallardo, A., Plaza, J., Plaza, A., 2017b. Cloud implementation of logistic regression for hyperspectral image classification. In: Vigo-Aguiar, J. (Ed.), Proceedings of the 17th International Conference on Computational and Mathematical Methods in Science and Engineering, CMMSE 2017. pp. 1063–2321.
He, H., Garcia, E.A., 2009. Learning from imbalanced data. IEEE Trans. Knowl. Data Eng. 21 (9), 1263–1284.
He, M., Li, X., Zhang, Y., Zhang, J., Wang, W., 2016. Hyperspectral image classification based on deep stacking network. In: 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS). pp. 3286–3289.
Hinton, G., Salakhutdinov, R., 2006. Reducing the dimensionality of data with neural networks. Science 313, 504–507.
Hinton, G.E., Osindero, S., Teh, Y.-W., 2006. A fast learning algorithm for deep belief nets. Neural Comput. 18 (7), 1527–1554.
Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R., 2012. Improving neural networks by preventing co-adaptation of feature detectors. CoRR. Available from: 1207.0580.
Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9 (8), 1735–1780.
Hu, F., Xia, G.-S., Hu, J., Zhang, L., Foody, G.M., Wang, L., Thenkabail, P.S., 2015a. Transferring deep convolutional neural networks for the scene classification of high-resolution remote sensing imagery in surveying, mapping and remote sensing. Remote Sens. 7 (11), 14680–14707 <http://www.mdpi.com/journal/remotesensing>.
Hu, W., Huang, Y., Wei, L., Zhang, F., Li, H., 2015b. Deep convolutional neural networks for hyperspectral image classification. J. Sensors.
Jarrett, K., Kavukcuoglu, K., Ranzato, M.A., Lecun, Y., 2009. What is the best multi-stage architecture for object recognition? In: ICCV. IEEE, pp. 2146–2153.
Karhunen, J., Raiko, T., Cho, K., 2015. Unsupervised Deep Learning: A Short Review.
Khodadadzadeh, M., Li, J., Plaza, A., Ghassemian, H., Bioucas-Dias, J.M., Li, X., 2014. Spectral-spatial classification of hyperspectral data using local and global probabilities for mixed pixel characterization. IEEE Trans. Geosci. Remote Sens. 52 (10), 6298–6314.
Kingma, D.P., Ba, J.L., 2014. ADAM: {A} method for stochastic optimization. CoRR. Available from: 1412.6980.
Kunkel, B., Blechinger, F., Lutz, R., Doerffer, R., van der Piepen, H., 1988. ROSIS (Reflective Optics System Imaging Spectrometer) – A candidate instrument for polar platform missions. In: Seeley, J., Bowyer, S. (Eds.), Optoelectronic technologies for remote sensing from space. pp. 134–141.
Larochelle, H., Bengio, Y., 2008. Classification using discriminative restricted boltzmann machines. In: Proceedings of the 25th International Conference on Machine learning – ICML '08, pp. 536.
LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521, 436–444.
LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., 1998a. Gradient-based learning applied to document recognition. Proc. IEEE.
LeCun, Y.A., Bottou, L., Orr, G.B., Müller, K.R., 1998b. Efficient backprop. In: Neural Networks: Tricks of the Trade, This Book is an Outgrowth of a 1996 NIPS Workshop. Springer-Verlag, pp. 9–50.
Li, J., Bioucas-Dias, J.M., Plaza, A., 2010. Semisupervised hyperspectral image segmentation using multinomial logistic regression with active learning. IEEE Trans. Geosci. Remote Sens. 48 (11), 4085–4098.
Li, J., Bioucas-Dias, J.M., Plaza, A., 2011. Hyperspectral image segmentation using a new Bayesian approach with active learning. IEEE Trans. Geosci. Remote Sens. 49 (10), 3947–3960.
Li, M., Zang, S., Zhang, B., Li, S., Wu, C., 2014a. A review of remote sensing image classification techniques: the role of spatio-contextual information. Eur. J. Remote Sens. 47, 389–411.
Li, T., Zhang, J., Zhang, Y., 2014b. Classification of hyperspectral image based on deep belief networks. In: Image Processing (ICIP), 2014 IEEE International Conference on. pp. 5132–5136.
Li, Y., Zhang, H., Shen, Q., 2017. Spectral-spatial classification of hyperspectral imagery with 3D convolutional neural network. Remote Sens. 9 (1), 67.
Licciardi, G.A., Del Frate, F., 2011. Pixel unmixing in hyperspectral data by means of neural networks. IEEE Trans. Geosci. Remote Sens. 49 (11), 4163–4172.
Liu, B., Yu, X., Zhang, P., Tan, X., Yu, A., Xue, Z., 2017. A semi-supervised convolutional neural network for hyperspectral image classification. Remote Sens. Lett. 8 (9), 839–848. https://doi.org/10.1080/2150704X.2017.1331053.
Liu, Q., Hang, R., Song, H., Zhu, F., Plaza, J., Plaza, A., 2016. Adaptive Deep Pyramid Matching for Remote Sensing Scene Classification. CoRR. Available from: 1611.03589.
Lu, X., Zheng, X., Yuan, Y., 2017. Remote sensing scene classification by unsupervised representation learning. IEEE Trans. Geosci. Remote Sens. (99), 1–10
Ma, X., Wang, H., Wang, J., 2016. Semisupervised classification for hyperspectral image based on multi-decision labeling and deep feature learning. ISPRS J. Photogramm. Remote Sens. 120, 99–107.
Melgani, F., Bruzzone, L., 2004. Classification of hyperspectral remote sensing images with support vector machines. IEEE Trans. Geosci. Remote Sens. 42 (8).
Midhun, E.M., Nair, S.R., Prabhakar, N., Kumar, S., 2014. Deep model for classification of hyperspectral image using restricted Boltzmann machine. In: Proceedings of the 2014 International Conference on Interdisciplinary Advances in Applied Computing. ACM, New York, NY, USA, pp. 35:1–35:7.
Nair, V., Hinton, G.E., 2010. Rectified linear units improve restricted Boltzmann machines. In: Fürnkranz, Johannes, Joachims, Thorsten (Eds.), Proceedings of the 27th International Conference on Machine Learning (ICML-10). Omnipress, pp. 807–814.
Paoletti, M.E., Haut, J.M., Plaza, J., Plaza, A., 2017. Yinyang K-means clustering for hyperspectral image analysis. In: Vigo-Aguiar, J. (Ed.), Proceedings of the 17th International Conference on Computational and Mathematical Methods in Science and Engineering. Rota, pp. 1625–1636.
Plaza, A., Benediktsson, J.A., Boardman, J.W., Brazile, J., Bruzzone, L., Camps-Valls, G., Chanussot, J., Fauvel, M., Gamba, P., Gualtieri, A., Marconcini, M., Tilton, J.C., Trianni, G., 2009. Recent advances in techniques for hyperspectral image processing. Remote Sens. Environ. 113 (1), S110–S122.
Qian, N., 1999. On the momentum term in gradient descent learning algorithms. Neural Netw. 12 (1), 145–151.
Rajan, S., Ghosh, J., Crawford, M.M., 2008. An active learning approach to hyperspectral data classification. IEEE Trans. Geosci. Remote Sens. 46 (4), 1231–1242.
Ranzato, M.A., Poultney, C., Chopra, S., Lecun, Y., 2006. Efficient learning of sparse representations with an energy-based model. In: Schölkopf, B., Platt, J., Hoffman, T. (Eds.), Advances in Neural Information Processing Systems, vol. 19. MIT Press, pp. 1137–1144.
Romero, A., Gatta, C., Camps-Valls, G., 2016. Unsupervised deep feature extraction for remote sensing image classification. IEEE Trans. Geosci. Remote Sens. 54 (3), 1349–1362.
Salakhutdinov, R., Hinton, G., 2009. Deep boltzmann machines. In: 12th International Conference on Artificial Intelligence and Statistics, pp. 3.
Scholkopf, B., Smola, A.J., 2001. Learning with Kernels:Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, Cambridge, MA, USA.
Smolensky, P., 1986. Information processing in dynamical systems: foundations of harmony theory. In: Rumelhart, David E., McLelland, J.L. (Eds.), Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Foundations, vol. 1. MIT Press, pp. 194–281 (Chapter 6).
Starck, J.-L., Elad, M., Donoho, D.L., 2005. Image decomposition via the combination of sparse representations and a variational approach. IEEE Trans. Image Process. 14 (10), 1570–1582.
Tarabalka, Y., Benediktsson, J.A., Chanussot, J., 2009. Spectral-spatial classification of hyperspectral imagery based on partitional clustering techniques. IEEE Trans. Geosci. Remote Sens. 47 (8), 2973–2987.
Vetrivel, A., Gerke, M., Kerle, N., Nex, F., Vosselman, G., 2018. Disaster damage detection through synergistic use of deep learning and 3D point cloud features derived from very high resolution oblique aerial images, and multiple-kernel-learning. ISPRS J. Photogramm. Remote Sens. 140, 45–59.
Wu, Q., Diao, W., Dou, F., Sun, X., Zheng, X., Fu, K., Zhao, F., 2016. Shape-based object extraction in high-resolution remote-sensing images using deep Boltzmann machine. Int. J. Remote Sens. 37 (24), 6012–6022.
Wu, Z., Wang, Q., Plaza, A., Li, J., Wei, Z., 2015. Real-time implementation of the sparse multinomial logistic regression for hyperspectral image classification on GPUs. IEEE Geosci. Remote Sens. Lett. 12 (7), 1456–1460.
Xu, X., Li, J., Huang, X., Dalla Mura, M., Plaza, A., 2016a. Multiple morphological component analysis based decomposition for remote sensing image classification. IEEE Trans. Geosci. Remote Sens. 54 (5), 3083–3102.
Xu, X., Lil, f., Plaza, A., 2016b. Fusion of hyperspectral and LiDAR data using morphological component analysis. In: 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 3575–3578.
Yang, H., 1999. A back-propagation neural network for mineralogical mapping from AVIRIS data. Int. J. Remote Sens. 20 (1), 97–110. https://doi.org/10.1080/014311699213622.
Yang, J., Zhao, Y., Chan, J.C.W., Yi, C., 2016. Hyperspectral image classification using two-channel deep convolutional neural network. In: 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 5079–5082.
Yu, S., Jia, S., Xu, C., 1, 2017. Convolutional neural networks for hyperspectral image classification. Neurocomputing 219, 88–98.
Yuan, Y., Mou, L., Lu, X., 2015. Scene recognition by manifold regularized deep learning architecture. IEEE Trans. Neural Netw. Learn. Syst. 26 (10), 2222–2233.

Zeiler, M.D., 2012. ADADELTA: An Adaptive Learning Rate Method. CoRR. Available from: <1212.5701>.

Zeiler, M.D., Krishnan, D., Taylor, G.W., Fergus, R., June 2010. Deconvolutional networks. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 2528–2535.

Zhang, H., Li, Y., Zhang, Y., Shen, Q., 2017. Spectral-spatial classification of hyperspectral imagery using a dual-channel convolutional neural network. Remote Sens. Lett. 8 (5), 438–447.

Zhang, L., Zhang, L., Du, B., 2016a. Deep learning for remote sensing data advances in machine learning for remote sensing and geosciences. IEEE Geosci. Remote Sens. Mag. 4 (2), 22–40 <http://ieeexplore.ieee.org/document/7486259/>.

Zhang, P., Gong, M., Su, L., Liu, J., Li, Z., 2016b. Change detection based on deep feature representation and mapping transformation for multi-spatial-resolution remote sensing images. ISPRS J. Photogramm. Remote Sens. 116, 24–41.

Zhao, W., Du, S., 2016a. Learning multiscale and deep representations for classifying remotely sensed imagery. ISPRS J. Photogramm. Remote Sens. 128, 223–239.

Zhao, W., Du, S., 2016b. Spectral-spatial feature extraction for hyperspectral image classification: a dimension reduction and deep learning approach. IEEE Trans. Geosci. Remote Sens. 54 (8), 4544–4554.

Zhou, X., Prasad, S., 2017. Active and semisupervised learning with morphological component analysis for hyperspectral image classification. IEEE Geosci. Remote Sens. Lett. 14 (8), 1348–1352.

Escuela Politecnica
Av. de la Universidad, S/N, 10003
Caceres, Spain
Phone: 0034927257000. Ext. 51662
Email: aplaza,jplaza{@unex.es}

Dr. Antonio Plaza Miguel y Dr. Javier Plaza Miguel como directores de la tesis titulada "Procesamiento eficiente y profundo de imgenes hiperespectrales de la observación remota de la Tierra y aplicaciones en tareas de clasificación", certifico el factor de impacto y la categorización de la siguiente publicación, incluida en la tesis doctoral. Del mismo modo, se especifica la aportación del doctorado. Si necesitas cualquier información o clarificación, por favor, no dude en contactar conmigo.

Antonio Plaza Miguel PhD. and Javier Plaza Miguel PhD as directors of the Phd thesis titled "Deep-efficient processing of remote sensing hyperspectral images and applications in classification tasks.", certify the impact factor and the categorization of the following publication, included in the doctoral thesis. In the same way, the contribution of the doctorate is specified. If you need any further information or clarification, please do not hesitate contacting me.

Artculo / Paper

Autores/Authors: **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza.
Title: Neighboring Region Dropout for Hyperspectral Image Classification.
Journal: IEEE Geoscience and Remote Sensing Letters.
Other Information: In press, 2020.
DOI: 10.1109/LGRS.2019.2940467.
Impact factor 2018: 3.534. Q1
Abstract: Deep neural networks (DNNs) exhibit great performance in the task of hyperspectral image (HSI) classification. However, these models are usually overparameterized and require large amounts of training data in order to properly avoid the curse of dimensionality and the variability of spectral signatures, thus suffering from overfitting problems when very few training samples are available, due to poor generalization ability in this particular case. The traditional regularization dropout (DO) strategy has been shown to be effective in fully connected DNNs but not in convolutional-based ones. This is mainly due to the way these architectures manage the spatial information. In this letter, we introduce a new approach to improve the generalization of convolutional-based models for HSI classification. Specifically, we develop a neighboring region DO technique that selectively cuts off certain neighboring outputs, creating spatial dropped regions. Our experimental results with two well-known HSIs reveal that the newly proposed method helps to achieve better classification accuracy than the traditional DO strategy, with a low computational cost.

Contribución del doctorado: Planteamiento de la hipótesis, desarrollo práctico, análisis y discusión de los resultados, elaboración y escritura del manuscrito.

Firma / Signature
Mayo / May, 2020

Antonio Plaza Miguel

Javier Plaza Miguel

Neighboring Region Dropout for Hyperspectral Image Classification

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

***Abstract*— Deep neural networks (DNNs) exhibit great performance in the task of hyperspectral image (HSI) classification. However, these models are usually overparameterized and require large amounts of training data in order to properly avoid the curse of dimensionality and the variability of spectral signatures, thus suffering from overfitting problems when very few training samples are available, due to poor generalization ability in this particular case. The traditional regularization dropout (DO) strategy has been shown to be effective in fully connected DNNs but not in convolutional-based ones. This is mainly due to the way these architectures manage the spatial information. In this letter, we introduce a new approach to improve the generalization of convolutional-based models for HSI classification. Specifically, we develop a neighboring region DO technique that selectively cuts off certain neighboring outputs, creating spatial dropped regions. Our experimental results with two well-known HSIs reveal that the newly proposed method helps to achieve better classification accuracy than the traditional DO strategy, with a low computational cost.**

***Index Terms*— Convolutional neural networks (CNNs), dropout (DO), hyperspectral images (HSIs), regularization.**

I. Introduction

HYPERSPECTRAL images (HSIs) comprise big cubes of adjacent spectral bands, where each pixel records the electromagnetic interaction between the incident solar radiation and the observed objects in a spectral signature that can be considered unique for each material on the surface of the earth. This allows a detailed characterization of observed areas and enables their successful exploitation on a wide range of applications [1].

The huge amount of information contained in HSI data cubes has been exploited by a large variety of spectral, spatial, and spectral–spatial classification methods, offering models with good performance in the task of understanding those features and relationship contained in the image. Traditionally, the following three kinds of methods have been established depending on the training procedure.

1) Unsupervised methods do not need to be trained, as they do not use labeled samples to fine-tune the model, being quite popular some clustering method such as k-means [2], linear discriminant analysis (LDA) [3], or probabilistic latent semantic analysis (PLSA) [4].
2) Supervised methods split the available data into labeled and unlabeled samples in order to perform the training and the inference steps. Some widely used supervised classifiers are the multinomial logistic regression (MLR) [5] or the support vector machine (SVM) [6].
3) Semisupervised methods apply different strategies to include unlabeled data, such as active learning approaches [7], or to expand the training set, using, for instance, generative adversarial networks (GANs) [8].

With the release of large and complex HSI data sets, the development of new classification algorithms is required in order to properly interpret the acquired data. Deep learning and convolutional-based approaches have been successfully used for this purpose, reaching excellent performance due to their inherent ability for exploiting different spectral and spatial features through deep and hierarchical architectures made up of stacked feature extractors [9], [10]. However, the performance of these methods for HSI classification is bounded by the high dimensionality of the data, the limited number of available labeled samples, and, generally, the low spatial resolution of HSIs, leading to the curse of dimensionality (Hughes effect), overfitting, and data variability problems [1].

Deep neural networks (DNNs), in general, and convolutional neural networks (CNNs) [11], in particular, can be seen as approximators of the form $f : \mathcal{X} \rightarrow \mathcal{Y}$ that solve an optimization function subject to a loss expression L. In this sense, supervised DNNs are mapping problems where, given an HSI data set, find the corresponding labels by tuning a parameterized model $\mathcal{M}(\mathcal{X}, \theta) = \mathcal{Y}$, whose parameters θ (distributed among the layers' stack) should minimize the error between the predicted and the expected outputs. Recent works claim that the deeper the $\mathcal{M}$, the better the accuracy that can be achieved [12]. This has a direct effect on the number of parameters, imposing severe restrictions on the amount of employed training samples, apart from the data degradation factor that is directly related to the increase of the model's depth. In this regard, several data augmenting and regularization techniques have been explored to avoid these problems. Focusing on the first ones, some works propose to increase the training set by applying slight spectral modifications and

Manuscript received March 14, 2019; revised July 14, 2019; accepted August 28, 2019. This work was supported in part by Spanish Ministry under Grant FPU14/02012-FPU15/02090 and Grant ESP2016-79503-C2-2-P, in part by Junta de Extremadura under Grant GR18060, and in part by European Union under Grant 734541-EOXPOSURE. *(Corresponding author: Mercedes E. Paoletti.)*

The authors are with the 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; jplaza@unex.es; aplaza@unex.es).

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

Digital Object Identifier 10.1109/LGRS.2019.2940467

spatial transformations to the available data [13], although these approaches are very time-consuming (simpler methods, such as the addition of random noise, do not take into account the spatial characteristics of the image). Haut *et al.* [14] introduced random occlusion (RO) as a new data augmenting method that drops certain areas on the CNN's inputs, maintaining the spatial consistency between the dropped zones. Although this approach exhibits good performance, it is not robust to network parameters, in the sense that relations between the weights of adjacent layers are not encouraged. On the other hand, regularization methods such as dropout (DO) [15] are able to strengthen the model, enforcing independence between adjacent layer weights by setting to zero some randomly selected neural activations during the training stage. In this context, DO is widely used due to its simplicity and low computational cost [9], being effective particularly on the fully connected layers of the CNN's architecture. However, its performance with convolutional layers is not that impressive, due to its random feature clipping, which does not take into account spatial implications. In fact, the effects of DO on a convolutional's output imitate the traditional "salt&pepper" noise, while feature maps remain spatially correlated. In the end, the extracted information is still propagated to the following layers [16].

In this context, inspired by the original DO mechanism and the RO data augmenting approach, this letter introduces the neighboring region DO (NRDO), a new spatially correlated DO mechanism in which random neighboring kernel's activations are dropped, creating occluded areas on the convolutional layers' output volumes that maintain spatial consistency while avoiding a significant increase in computational complexity [16]. The remainder of this letter is organized as follows. Section II describes the proposed method. Section III discusses the performance of the proposed NRDO using two HSIs, demonstrating its accuracy. Section IV concludes with some remarks and hints at plausible future research lines.

II. Methodology

Let us define $\mathbf{X} \in \mathbb{R}^{n_1 \times n_2 \times n_{\text{bands}}}$ as a HSI cube, where $n_1 \times n_2$ are the spatial components and n_{bands} is the number of spectral bands. Each HSI pixel can be represented as a spectral vector $\mathbf{x}_{i,j} \in \mathbb{R}^{n_{\text{bands}}}$. An end-to-end spatial 2-DCNN model normally performs a preprocessing step, encoding the spectral information into one band by using, for instance, principal component analysis (PCA) and extracts, for each 1-D pixel $x_{i,j}$, a neighboring window $\mathbf{p}_{i,j} \in \mathbb{R}^{d \times d \times 1}$ of $d \times d$ spatial dimensions, with $x_{i,j}$ being the central pixel. During the training stage, pairs of patches and labels $\{\mathbf{p}_{i,j}, \mathbf{y}_{i,j}\}$ are used to create the training set, where $\mathbf{y}_{i,j} \in \mathbb{R}^{n_{\text{classes}}}$ represents the label of the (i, j)th patch's central pixel in one-hot encoding way. These patches are fed to the 2-DCNN, which applies a hierarchical stack of feature extraction (FE) stages to obtain different levels of data representation, until reaching an abstract representation at the end, that encodes the more descriptive features and internal nonlinear data relationship, which are employed by the final classifier layers to produce a classification output.

Each FE-stage is usually composed by a set of different layers, being the convolutional layer the major responsable for the extraction. Each layer l defines $K^{(l)}$ filters, with $k^{(l)} \times k^{(l)}$ neurons each. In this sense, the kernel defined by the lth layer computes the operation over the input with sliding-step s, being overlapped on local areas. At the end, the kernel performs the linear convolution $(*)$ between the weights of the neurons $\mathbf{W}^{(l)}$, the input data volume $\mathbf{X}^{(l-1)}$, and the bias $b^{(l)}$, obtaining an output volume $\mathbf{X}^{(l)} \in \mathbb{R}^{n^{(l)} \times n^{(l)} \times K^{(l)}}$ of K feature maps with $n^{(l)} \times n^{(l)}$ extracted features

$$\mathbf{X}^{(l)} = \mathcal{H}(\mathbf{W}^{(l)} * \mathbf{X}^{(l-1)} + b^{(l)})_{K^{(l)} \times k^{(l)} \times k^{(l)}}. \tag{1}$$

After the FE-stage performed by the convolutional layer, a nonlinear activation function $\mathcal{H}(\cdot)$ is applied to the output volume in order to extract the nonlinear features and relationship contained into the volume. In our case, we apply the well-known rectified linear unit (ReLU) function. Also, a downsampling operation (implemented by max or average pooling) is applied in order to reduce the spatial dimensions and summarize the obtained features.

From (1), it can be observed that the outputs of previous layers are refined by the following ones, i.e., the CNN's neurons are, in fact, working in a cooperative way [15], [17]. Although this hierarchical mechanism is appealing during the training stage, introduce weak links between the neurons of adjacent layers, and hampering the inference step [9]. In this context, traditional DO [15] is applied between the activation and pooling layers as a regularization method to avoid overfitting and provide some independence between adjacent layers' neurons, by setting to zero some randomly selected neural activations. This improves the backpropagation procedure, where neurons should be adjusted in an individual way, instead of establishing trivial dependencies with other neurons. The main motivation behind this approach is to force the layer's neurons to extract more robust and discriminatory features on their own. Mathematically, we can break down (1) in order to focus on the (i, j)th extracted feature of convolutional layer l in its zth filter (with $z = \{1, \ldots, K^{(l)}\}$), to which a gating 0-1 Bernoulli variable is applied as the DO regularization term $\delta_{i,j}^{(l)}$, following a probability percentage $p^{(l)}$ which is fixed to the lth layer [18]:

$$\delta_{i,j}^{(l)} = \text{Bernoulli}(p^{(l)}) \tag{2a}$$

$$x_{i,j}^{(l)z} = \mathcal{H}\left(\delta_{i,j}^{(l)} \sum_{\hat{i}=1}^{k^{(l)}} \sum_{\hat{j}=1}^{k^{(l)}} \left(w_{\hat{i},\hat{j}}^{(l)} x_{(is+\hat{i}),(js+\hat{j})}^{(l-1)}\right) + b^{(l)}\right)_z. \tag{2b}$$

Fig. 1 provides a graphical illustration of how the DO regularization method works, using a synthetic feature map given in Fig. 1(a). As it can be observed in Fig. 1(b) and (c), the DO injects random noise to the feature maps in order to disentangle the behavior of adjacent layers' neurons. However, this noise is not structured, which makes it not completely effective in the task of removing semantic information of the feature map, where nearby features still contain related spatial information.

To overcome the limitations of the traditional DO strategy, we propose to inject spatial-structured noise at every feature map by dropping the output of neighboring neural activations, obtaining full-dropped spatial regions on the output feature maps that effectively remove spatial-correlated information. In this sense, the amount of neural activations $\gamma^{(l)}$ that will be dropped, coupled with the surrounding window's spatial size

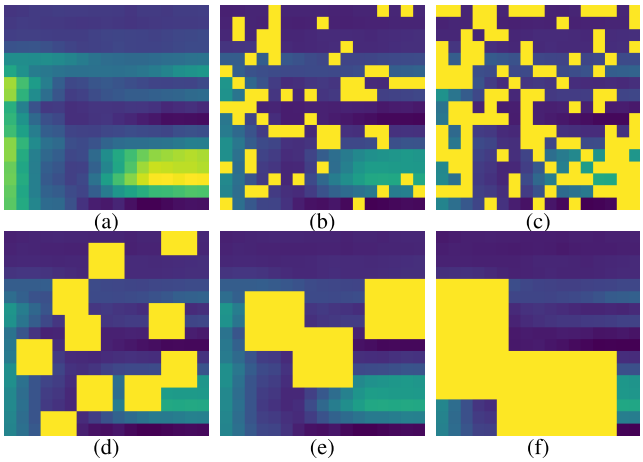


Fig. 1. Visualization of the original DO and the proposed NRDO performance over a feature map of size 17×17. The first row shows (a) original feature map and the feature maps obtained after dropping isolated samples using a DO of (b) $p^{(l)} = 20\%$ and (c) $p^{(l)} = 40\%$. The second row shows the feature map obtained after applying NRDO by configuring the dropping percentage and the dropping windows size to (d) $p^{(l)} = 20\%$ and $d^{(l)} = 3$, (e) $p^{(l)} = 20\%$ and $d^{(l)} = 5$, and (f) $p^{(l)} = 20\%$ with $d^{(l)} = 10$.

$d^{(l)}$ (that will be set to zero), must be defined at each layer l. Following the DO method, for each position (i, j) of the input feature map $\mathbf{X}^{(l-1)}$, our NRDO applies the gating variable $\delta_{i,j}^{(l)}$, obtained by the Bernoulli distribution with probability $\gamma^{(l)}$. In addition, for the zero variables $\delta_{i,j}^{(l)}$, a spatial square patch centered on (i, j) is obtained as a zero-mask with dimensions $d^{(l)} \times d^{(l)}$. Finally, this mask is overlapped and applied over the input volume $\mathbf{X}^{(l-1)}$, dropping the corresponding window in all the K filters of the lth layer. However, instead of setting a direct dropping probability, $\gamma^{(l)}$ is obtained as a correction of the traditional DO percentage $p^{(l)}$, the dropping window's size $d^{(l)}$, and the spatial dimension of the obtained feature map $\mathbf{X}^{(l)} \in \mathbb{R}^{n^{(l)} \times n^{(l)} \times K^{(l)}}$ by the lth convolutional layer. In this context, $\gamma^{(l)}$ is obtained as

$$\gamma^{(l)} = \frac{p^{(l)}}{(d^{(l)})^2} \frac{(n^{(l)})^2}{(n^{(l)} - d^{(l)} + 1)^2}. \tag{3}$$

It is recommened that $d^{(l)}$ is not be greater than $n^{(l)}$. In fact, (3) makes an approximation between the desired amount of dropped data, indicated by the known $p^{(l)}$, and the dropped neighborhood for each zero variable $\delta_{i,j}^{(l)}$, to make an equitable balance between the pixels and their surrounding windows to be dropped and the desired amount of spatial-structured noise to be injected.

Algorithm 1 gives a general overview of the proposed NRDO method, which is applied between the convolutional and nonlinear activation layers, following the scheme given in (2). An interesting aspect is the computation of the window to be dropped. As some HSI data sets are characterized by their low spatial resolution, our strategy can help in this particular case since the dropped neighborhoods are adapted to the feature map's margins, taking advantage of all the available features as we can observe in Fig. 1(d)–(f), where the dropped neighborhoods have been adjusted to the feature's edges. Moreover, these dropped windows can be also spatially overlapped, as it can be observed in Fig. 1(f),

Algorithm 1 NRDO

procedure NRDO($\mathbf{X}^{(l)} \in \mathbb{R}^{n^{(l)} \times n^{(l)} \times K^{(l)}}$: obtained feature map from l-th layer, $p^{(l)}$: dropping percentage, $d^{(l)}$: dropping window's size)
  $\gamma^{(l)} = \frac{p^{(l)}}{(d^{(l)})^2} \frac{(n^{(l)})^2}{(n^{(l)} - d^{(l)} + 1)^2}$ ▷ Dropping probability
  $\mathbf{M} = \text{Ones}(n^{(l)}, n^{(l)}, K^{(l)})$ ▷ Initializing mask
  for i, j in $n^{(l)}$ **do**
    if $\left(\delta_{i,j}^{(l)} = \text{Bernoulli}(\gamma^{(l)})\right) == 0$ **then**
      $\mathbf{M} = \text{Dropped\_Window\_on\_M}(i, j, d^{(l)})$ ▷ For each zero $\delta_{i,j}^{(l)}$, a zeroed square window $d^{(l)} \times d^{(l)}$ is set on the mask $\mathbf{M}$ with the center on the (i, j) position
    end if
  end for
  $\hat{\mathbf{X}}^{(l)} = \mathbf{M} \cdot \mathbf{X}^{(l)}$
  return $\hat{\mathbf{X}}^{(l)}$
end procedure

TABLE I
ARCHITECTURAL DETAILS OF THE PROPOSED MODEL

Layer ID	Kernel/Neurons	DO/NRDO	Max Pooling	Act. function
Conv1	$50 \times 3 \times 3 \times 1$	No	No	ReLU
Conv2	$100 \times 5 \times 5 \times 50$	Yes	2×2	ReLU
Conv3	$200 \times 5 \times 5 \times 100$	Yes	2×2	ReLU
Conv4	$400 \times 2 \times 2 \times 200$	No	No	ReLU
FC1	300	No	-	ReLU
FC2	$n_{classes}$	No	-	Softmax

where two dropped regions that are slightly overlapped can be appreciated.

On the other hand, the application of a rigorous NRDO with a fixed value of $p^{(l)}$ can negatively affect the performance of the network, while the implementation of a soft NRDO may not provide the desired robustness. In order to overcome the limitations, our model is trained with a $p^{(l)}$ whose value increases linearly and progressively through the epochs [16], from zero probability to the maximum indicated value of $p^{(l)}$, with the goal of progressively adapting the performance, extracting more robust and independent features at each epoch.

III. EXPERIMENTS

A. Experimental Configuration and Data Sets

In order to test the performance of the proposed regularization technique, a deep 2-DCNN model has been implemented for HSI classification. Inspired by previous works in [14], the proposed network is composed of four convolutional layers and two fully connected layers. Focusing on the convolutional layers, the second and third layers implement one of the two available dropping mechanism, DO or NRDO, for comparative purposes. Table I describes the details of the configuration of the network, which has been executed on a hardware environment composed by a sixth-generation Intel Core i7-6700K processor with 8M of Cache and up to 4.20 GHz (four cores/eight way multi-task processing), an ASUS Z170 progamming motherboard, a GPU NVIDIA GeForce GTX 1080 with 8-GB GDDR5X of video memory and 10 Gbps of memory frequency, 40 GB of DDR4 RAM with a serial speed of 2400 MHz and a Toshiba DT01ACA

TABLE II

COMPARISON BETWEEN DO AND NRDO, WITH DIFFERENT PERCENTAGES OF $p^{(l)}$ AND SETTING $d^{(l)} = 3$

	T. Size	2DCNN	*2DCNN + Non-Spatially Structured Dropout*			*2DCNN + Neighboring Region Dropout*		
			20%	40%	80%	20%	40%	80%
INDIAN P.	1%	52.85 ±1.90	52.69 ±2.18	53.61 ±1.40	56.84 ±1.16	56.24 ±1.08	56.72 ±2.63	**59.64** ±1.41
	3%	70.97 ±1.57	74.70 ±1.29	75.10 ±1.48	80.44 ±1.83	79.08 ±0.98	82.93 ±0.69	**85.81** ±1.05
	5%	81.45 ±1.24	84.40 ±2.18	86.46 ±1.33	90.39 ±0.78	89.61 ±1.05	91.58 ±0.84	**93.78** ±0.53
	10%	92.43 ±1.03	94.98 ±0.72	96.35 ±0.23	97.90 ±0.26	97.52 ±0.19	97.85 ±0.51	**98.49** ±0.29
	15%	96.42 ±1.04	98.18 ±0.10	98.62 ±0.30	99.34 ±0.17	99.03 ±0.21	99.24 ±0.23	**99.53** ±0.15
	20%	98.13 ±0.42	99.20 ±0.32	99.25 ±0.18	99.69 ±0.16	99.51 ±0.08	99.69 ±0.05	**99.83** ±0.06
U. PAVIA	1%	87.47 ±0.54	87.55 ±0.97	87.49 ±0.87	90.75 ±0.89	89.94 ±0.47	91.92 ±0.39	**92.39** ±0.54
	3%	95.62 ±0.41	95.56 ±0.37	96.42 ±0.21	98.14 ±0.16	97.35 ±0.33	97.90 ±0.21	**98.70** ±0.17
	5%	97.64 ±0.20	98.18 ±0.07	98.50 ±0.26	99.16 ±0.16	99.07 ±0.20	99.38 ±0.08	**99.57** ±0.08
	10%	99.20 ±0.14	99.51 ±0.08	99.62 ±0.08	99.82 ±0.05	99.79 ±0.07	99.89 ±0.02	**99.93** ±0.02
	15%	99.57 ±0.11	99.84 ±0.05	99.89 ±0.01	99.94 ±0.02	99.93 ±0.01	99.96 ±0.01	**99.98** ±0.01
	20%	99.81 ±0.02	99.92 ±0.01	99.95 ±0.02	99.98 ±0.02	99.98 ±0.01	99.99 ±0.01	99.99 ±0.01

TABLE III

OBTAINED OA RESULTS FOR EACH CONSIDERED CLASSIFIER

	T. Size	RF	SVM RBF	MLP	ELM	K-ELM	1DCNN	2DCNN			
								RAW	RO	DO	NRDO
INDIAN P.	1%	-	-	-	-	-	-	52.85	54.90	56.84	**59.64**
	3%	-	-	-	-	-	-	70.97	78.74	80.44	**85.81**
	5%	69.00	74.52	77.13	72.23	80.38	75.37	81.45	89.46	90.39	**93.78**
	10%	75.58	81.00	83.10	78.88	84.85	82.66	92.43	96.96	97.90	**98.49**
	15%	78.19	83.91	85.28	81.59	86.81	86.13	96.42	98.45	99.34	**99.53**
PAVIA U.	1%	81.35	88.91	88.11	80.28	82.05	85.70	87.47	88.69	90.75	**92.39**
	3%	-	-	-	-	-	-	95.62	96.94	98.14	**98.70**
	5%	87.36	93.43	93.19	85.35	87.07	91.01	97.64	98.64	99.16	**99.57**
	10%	89.51	94.45	94.36	86.72	88.69	94.13	99.20	99.68	99.82	**99.93**
	15%	90.48	94.89	94.91	87.24	89.41	95.29	99.57	99.92	99.94	**99.98**

HDD with 7200RPM and 2 TB of storage capacity. In addition, and in order to efficiently implement the proposed approach, it has been parallelized over the GPU using CUDA language over Pytorch framework. Finally, all the codes and examples presented in this letter are available online.[1] The proposed method has been tested over two widely used HSI data sets. The first one is the AVIRIS's Indian Pines (IP) scene, which has 145 × 145 samples with low spatial resolution of 20mpp and 200 spectral bands in the wavelength range from 0.4 to 2.5 μm. It was captured over an agricultural and forest area and its ground truth is composed of 16 different classes. The second one is the ROSIS's University of Pavia (UP) scene, which contains 610 × 340 samples with higher (1.3 mpp) spatial resolution and 113 spectral bands in the wavelength range from 0.43 to 0.86 μm. It was captured over an urban area and its ground truth is composed of nine different classes.

B. Experimental Results and Discussion

1) Comparison Between Dropout and Neighboring Region Dropout: First, experiment compares the performance of the 2-DCNN model with regularization method, considering the original nonspatially structured DO and the proposed NRDO. Each model has been trained over IP and UP with 1%, 3%, 5%, 10%, 15%, and 20% of randomly selected samples, input patch size of 11 × 11, and different dropping percentages ($p^{(l)} = \{20\%, 40\%, 80\%\}$), fixing the dropping window's size to $d^{(l)} = 3$ in the case of NRDO. Table II shows the obtained results. Focusing on DO, this strategy is highly beneficial when the scene is spectrally mixed and contains few regular spatial structures (as it is the case with the IP scene). Moreover, the bigger $p^{(l)}$, the larger the overall accuracy (OA) improvement. However, with the UP scene, the effectiveness of DO is appreciably lower than with the IP scene (in fact, only in the case of $p^{(l)} = 80\%$, the OA values rise by more than 1% point for small training sets), even reducing the overall performance with limited training samples (1% of IP and 3% of UP employing $p^{(l)} = 20\%$). In this sense, the proposed method exhibits a more consistent behavior with both data sets, being able to outperform the results obtained by DO and significantly improving the results obtained by the original 2-DCNN without regularization method and exhibiting a lower standard deviation. The effectiveness of this method is visibly high in IP and UP, in particular, when small training sets are considered, reaching the best OA performances when $p^{(l)} = 80\%$. It must be noted that, since NRDO occludes entire windows, it prevents the model from seeing all the complete features of the input data, forcing the network to look for more robust parameters.

2) Comparison Between Neighboring Region Dropout and Several Classifiers: Second, experiment performs a comparison between the proposal NRDO, with $p^{(l)} = 40\%$ and $d^{(l)} = 3$, and six different classifiers: 1) random forest (RF); 2) SVM with radial basis function; 3) shallow multilayer perceptron (MLP); 4) basic and kernel extreme learning machines (ELM and K-ELM); 5) spectral 1-DCNN; and 6) spatial 2-DCNN. In addition, four 2-DCNN models have been considered: without data augmenting or DO methods (original data), with RO [14], with $p^{(l)} = 80\%$ of DO, and with

[1] https://github.com/mhaut/DeepNRD

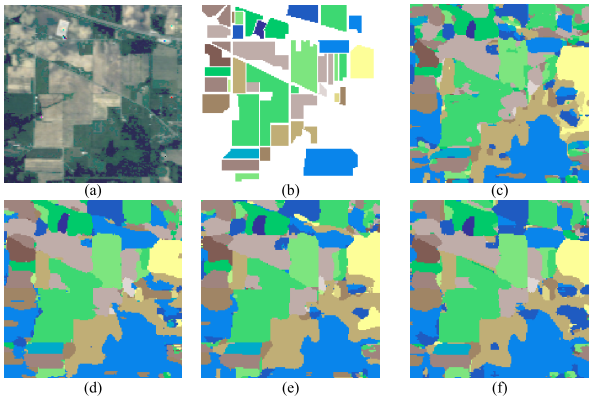


Fig. 2. Classification maps of IP, being (a) simulated RGB composition of the scene, (b) ground truth, and from (c) to (f), the obtained classification maps corresponding to the 2-DCNN models of Table III. (a) RGB. (b) GT. (c) 2-DCNN (92.43%). (d) 2-DCNN-RO% (96.96%). (e) 2-DCNN-DO% (97.90%). (f) 2-DCNN-NRDO% (98.49%).

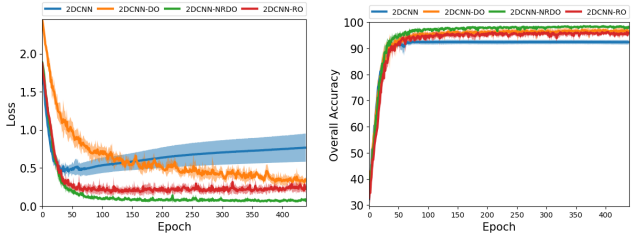


Fig. 3. Evolution of the (Left) loss and (Right) OA as a function of the number of training epochs when using the 2-DCNN with and without DO and NRDO regularization methods and RO data augmenting method, over IP data set, with 10% of training data and setting $p^{(l)} = 80\%$ and $d^{(l)} = 3\%$.

$p^{(l)} = 80\%$ of NRDO. Table III reveals that spatial models are able to greatly outperform spectral methods, reaching 90% and 99% of OA when classifying IP and UP scenes, respectively. Focusing in spatial models, the proposed NRDO is able to reduce the overfitting problem when lower training percentages are employed, achieving the best result in all the experiments. This suggests that neurons are able to learn independently while retaining a spatial context, so the final classification becomes more robust. Fig. 2 shows classification maps obtained by the spatial classifiers. It can be seen that the proposed method is able to correctly classify even the smallest and more complex classes, thus providing a more detailed map. Finally, Fig. 3 shows the evolution of the loss and OA with increasing epochs obtained by the 2-DCNN without any data augmenting/regularization method, and with RO, DO, and NRDO. Looking at the raw model, the loss grows as the epochs increase, indicating a clear overfitting problem. Although the RO decreases the loss faster than DO, it also suffers the overfitting in the final epochs. However, the proposed method is able to achieve lower and stable loss scores than DO and RO. This is also observed in the evolution of OA, where the NRDO enables a better tuning of the result.

IV. Conclusion

This letter evaluates a spatial-structured regularization technique for HSI data classification, which is based on randomly drop squared-windows of the convolutional-extracted feature maps, retaining the spatial consistency and allowing a strongest, deep and independent learning of the layer's neurons. Obtained results demonstrate that not only the proposed approach efficiently deals with the overfitting problem when low training data are available but is also able to reach a better performance than other compared techniques. Moreover, as the proposal improves the performance of the convolutional layer, it can be effectively used in more complex models, such as ResNets and DenseNets. Finally, since the approach is not restricted to spatial classifiers, in the future, we plan to incorporate it to spatial–spectral models too.

References

[1] M. E. Paoletti *et al.*, "Capsule networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 4, pp. 2145–2160, Apr. 2019.
[2] J. Haut, M. Paoletti, J. Plaza, and A. Plaza, "Cloud implementation of the K-means algorithm for hyperspectral image analysis," *J. Supercomput.*, vol. 73, no. 1, pp. 514–529, 2017.
[3] R. Bahmanyar, D. Espinoza-Molina, and M. Datcu, "Multisensor earth observation image classification based on a multimodal latent Dirichlet allocation model," *IEEE Geosci. Remote Sens. Lett.*, vol. 15, no. 3, pp. 459–463, Mar. 2018.
[4] R. Fernandez-Beltran, J. M. Haut, M. E. Paoletti, J. Plaza, A. Plaza, and F. Pla, "Remote sensing image fusion using hierarchical multimodal probabilistic latent semantic analysis," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 11, no. 12, pp. 4982–4993, Dec. 2018.
[5] M. Khodadadzadeh, J. Li, A. Plaza, and J. M. Bioucas-Dias, "A subspace-based multinomial logistic regression for hyperspectral image classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 11, no. 12, pp. 2105–2109, Dec. 2014.
[6] Z. Shao, L. Zhang, X. Zhou, and L. Ding, "A novel hierarchical semisupervised SVM for classification of hyperspectral images," *IEEE Geosci. Remote Sens. Lett.*, vol. 11, no. 9, pp. 1609–1613, Sep. 2014.
[7] 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.
[8] L. Zhu, Y. Chen, P. Ghamisi, and J. A. Benediktsson, "Generative adversarial networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 9, pp. 5046–5063, Sep. 2018.
[9] M. E. Paoletti, J. M. Haut, J. Plaza, and A. Plaza, "A new deep convolutional neural network for fast hyperspectral image classification," *ISPRS J. Photogram. Remote Sens.*, vol. 145, Part A, pp. 120–147.
[10] L. Zhang, Q. Zhang, B. Du, X. Huang, Y. Y. Tang, and D. Tao, "Simultaneous spectral-spatial feature selection and extraction for hyperspectral images," *IEEE Trans. Cybern.*, vol. 48, no. 1, pp. 16–28, Jan. 2018.
[11] L. Zhang, L. Zhang, and B. Du, "Deep learning for remote sensing data: A technical tutorial on the state of the Art," *IEEE Geosci. Remote Sens. Mag.*, vol. 4, no. 2, pp. 22–40, Jun. 2016.
[12] 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, Feb. 2019.
[13] W. Li, C. Chen, M. Zhang, H. Li, and Q. Du, "Data augmentation for hyperspectral image classification with deep CNN," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 4, pp. 593–597, Apr. 2019.
[14] J. M. Haut, M. E. Paoletti, J. Plaza, J. Li, and A. Plaza, "Hyperspectral image classification using random occlusion data augmentation," *IEEE Geosci. Remote Sens. Lett.*, to be published.
[15] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.
[16] G. Ghiasi, T.-Y. Lin, and Q. V. Le, "Dropblock: A regularization method for convolutional networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 10750–10760.
[17] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," Jul. 2012, *arXiv:1207.0580*. [Online]. Available:https://arxiv.org/abs/1207.0580
[18] P. Baldi and P. J. Sadowski, "Understanding dropout," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2814–2822.

Escuela Politecnica
Av. de la Universidad, S/N, 10003
Caceres, Spain
Phone: 0034927257000. Ext. 51662
Email: aplaza,jplaza{@unex.es}

Dr. Antonio Plaza Miguel y Dr. Javier Plaza Miguel como directores de la tesis titulada "Procesamiento eficiente y profundo de imgenes hiperespectrales de la observación remota de la Tierra y aplicaciones en tareas de clasificación", certifico el factor de impacto y la categorización de la siguiente publicación, incluida en la tesis doctoral. Del mismo modo, se especifica la aportación del doctorado. Si necesitas cualquier información o clarificación, por favor, no dude en contactar conmigo.

Antonio Plaza Miguel PhD. and Javier Plaza Miguel PhD as directors of the Phd thesis titled "Deep-efficient processing of remote sensing hyperspectral images and applications in classification tasks.", certify the impact factor and the categorization of the following publication, included in the doctoral thesis. In the same way, the contribution of the doctorate is specified. If you need any further information or clarification, please do not hesitate contacting me.

Artculo / Paper

Autores/Authors: J. M. Haut, **M. E. Paoletti**, J. Plaza, A. Plaza and J. Li
Title: Visual Attention-Driven Hyperspectral Image Classification.
Journal: IEEE Transactions on Geoscience and Remote Sensing.
Other Information: vol. 57, no. 10, pp. 8065-8080, October 2019.
DOI: 10.1109/TGRS.2019.2918080.
Impact factor 2018: 5.630. Q1
Abstract: Deep neural networks (DNNs), including convolutional neural networks (CNNs) and residual networks (ResNets) models, are able to learn abstract representations from the input data by considering a deep hierarchy of layers that perform advanced feature extraction. The combination of these models with visual attention techniques can assist with the identification of the most representative parts of the data from a visual standpoint, obtained through more detailed filtering of the features extracted by the operational layers of the network. This is of significant interest for analyzing remotely sensed hyperspectral images (HSIs), characterized by their very high spectral dimensionality. However, few efforts have been conducted in the literature in order to adapt visual attention methods to remotely sensed HSI data analysis. In this paper, we introduce a new visual attention-driven technique for the HSI classification. Specifically, we incorporate attention mechanisms to a ResNet in order to better characterize the spectral-spatial information contained in the data. Our newly proposed method calculates a mask that is applied to the features obtained by the network in order to identify the most desirable ones for classification purposes. Our experiments, conducted using four widely used HSI data sets, reveal that the proposed deep attention model provides competitive advantages in terms of classification accuracy when compared to other state-of-the-art methods.

Contribución del doctorado: Planteamiento de la hipótesis, desarrollo práctico, análisis y discusión de los resultados, elaboración y escritura del manuscrito.

Firma / Signature
Mayo / May, 2020

Antonio Plaza Miguel

Javier Plaza Miguel

Visual Attention-Driven Hyperspectral Image Classification

Juan Mario Haut, *Student Member, IEEE*, Mercedes E. Paoletti, *Student Member, IEEE*, Javier Plaza, *Senior Member, IEEE*, Antonio Plaza, *Fellow, IEEE*, and Jun Li, *Senior Member, IEEE*

***Abstract*— Deep neural networks (DNNs), including convolutional neural networks (CNNs) and residual networks (ResNets) models, are able to learn abstract representations from the input data by considering a deep hierarchy of layers that perform advanced feature extraction. The combination of these models with visual attention techniques can assist with the identification of the most representative parts of the data from a visual standpoint, obtained through more detailed filtering of the features extracted by the operational layers of the network. This is of significant interest for analyzing remotely sensed hyperspectral images (HSIs), characterized by their very high spectral dimensionality. However, few efforts have been conducted in the literature in order to adapt visual attention methods to remotely sensed HSI data analysis. In this paper, we introduce a new visual attention-driven technique for the HSI classification. Specifically, we incorporate attention mechanisms to a ResNet in order to better characterize the spectral–spatial information contained in the data. Our newly proposed method calculates a mask that is applied to the features obtained by the network in order to identify the most desirable ones for classification purposes. Our experiments, conducted using four widely used HSI data sets, reveal that the proposed deep attention model provides competitive advantages in terms of classification accuracy when compared to other state-of-the-art methods.**

Manuscript received November 18, 2018; revised April 15, 2019; accepted May 16, 2019. Date of publication June 12, 2019; date of current version September 25, 2019. This work was supported in part by the 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, in part supported by 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) under Grant GR18060, in part by the European Union's Horizon 2020 Research And Innovation Programme under Grant Agreement 734541 (EOXPOSURE), in part by the National Natural Science Foundation of China under Grant 61771496, in part by the Guangdong Provincial Natural Science Foundation under Grant 2016A030313254, and in part by the National Key Research and Development Program of China under Grant 2017YFB0502900. *(Corresponding author: Jun Li.)*

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

J. Li is with the Guangdong Provincial Key Laboratory of Urbanization and Geosimulation, Center of Integrated Geographic Information Analysis, School of Geography and Planning, Sun Yat-sen University, Guangzhou 510275, China (e-mail: lijun48@mail.sysu.edu.cn).

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.2019.2918080

***Index Terms*— Deep learning (DL), feature extraction, hyperspectral image (HSI) classification, residual neural networks, visual attention.**

I. Introduction

HYPERSPECTRAL image (HSI) classification is a very active research field in remote sensing and earth observation [1], [2]. This is due to the excellent characterization that HSI instruments can provide for large areas on the surface of the earth. HSI data are often collected by imaging spectrometers mounted on aerial or satellite platforms and comprise hundreds of images at different (continuous and narrow) wavelengths, usually from the visible to the near-infrared regions of the electromagnetic spectrum. As a result, high-dimensional data cubes are obtained, in which each pixel captures the emitted, reflected, and transmitted light over the observed land cover materials. Each pixel (vector) in the data cube can be interpreted as a spectral signature or *fingerprint* that uniquely characterizes the observed materials of the target area [3]. Such data cubes provide a wealth of spectral and spatial information, a property that is very useful for monitoring the surface of the earth [4], [5] in a wide range of applications, such as precision agriculture [6]–[8], environmental and natural resources management [9], surveillance [10]–[12], and others [13].

HSI classification has been usually tackled as an optimization problem, trying to assign each pixel of the scene to a certain land cover class by adapting traditional image analysis methods to HSI data [14]. For instance, standard machine learning methods assume that the HSI data cube is a collection of spectral vectors with no spatial arrangement, exploiting only the spectral information to discriminate and classify the pixels. Several unsupervised and supervised spectral-based approaches have been applied to interpret the HSI data, including k-means clustering [15], k-nearest neighbors (KNNs) [16], support vector machines (SVMs) [17], [18] and other kernel-based methods [19], [20], logistic regression (LR) [21], or random forest (RF) [22], among many others. However, the classification of HSI data involves certain difficulties not to be found in other kinds of image data (in addition to the huge amount of information contained in HSI data cubes [2]). Specifically, traditional supervised classification approaches are largely affected by the *curse of dimensionality* [23], which may hamper the accuracy of the classifier when the number of available labeled training samples is limited in

relation to the (high) dimensionality of the data. This is also due to the high cost and effort involved in expert annotation of labeled data, a fact that can result in an undercomplete training process that is prone to overfitting (this is also known as the Hughes phenomenon [24]). Moreover, HSI data sets present high intraclass variability and interclass similarity, resulting from atmospheric interferers, spectral variability, and the configuration of the sensor. These aspects bring additional difficulties when characterizing the data and call for new techniques that can better exploit the rich spatial and spectral information contained in HSI scenes.

To address some of the aforementioned issues, several deep neural network (DNN) models have been developed in the literature [25]. These flexible architectures, composed by a stack of layers, allow multiple techniques to include and process not only the spectral signatures but also the spatial-contextual information contained in the captured scenes. Based on the idea that spatially adjacent pixels often belong to the same class, these classifiers take advantage of the spatial information to reduce sample variability. In fact, it is well-known that the extraction of spectral–spatial features is very useful to improve the classification process, helping to reduce label uncertainty and intraclass variance. As a result, joint spectral–spatial methods can often perform better than purely spectral- or spatial-based ones. However, in deep learning (DL) methods, there is a problem of how to fuse the spectral and spatial information. Focusing on stacked autoencoders (SAEs) [26] and deep belief networks (DBNs) [27], we can find several techniques that concatenate the spectral signatures and the spatial information extracted from neighboring pixels by taking advantage of simple dimensionality reduction methods, such as the principal component analysis (PCA) [28]–[31] or more sophisticated methods, such as superpixels [32], guided filtering [33], or morphological profiles [34], [35], among others. Traditional fully connected architectures admit vector inputs, so the spatial structure is usually lost. In this sense, convolutional neural networks (CNNs) [36] are the powerful tool for the analysis of HSI images due to their capacity to accurately characterize both the spectral- and spatial-contextual information contained in HSI data cubes [37], being able to effectively extract the features with a high-level of abstraction from the raw data and achieving excellent classification results [38].

However, DL-based models are not totally immune to the curse of dimensionality and the Hughes phenomenon. In fact, CNNs tend to quickly overfit when a few labeled samples are available. To overcome this limitation, several techniques have been developed, including: 1) semisupervised and active learning (AL) techniques [39], able to deal with overfitting when very few training samples are available; 2) residual learning [e.g., using residual networks (ResNets)] [40], [41] and dense connections [e.g., using dense networks (DenseNets)] [42], [43], which can alleviate the loss of information and vanishing gradient problems of very deep and complex architectures; and 3) the development of new information routing techniques, such as capsule modules [e.g., using capsule networks (CapsNets)] [44], [45]. Despite these advances, CNN-based models still present the main limitation when dealing with HSI data. In fact, they can be hindered by the mode operation of their own convolution filters that treat the input content completely equally, while probably not all spectral–spatial information provided by the input hyperspectral pixels are equally interesting, informative, relevant, and/or predictive for classification purposes [46].

In the area of computer vision, several efforts are now being made to improve DL techniques, overcoming the equal treatment of the convolution kernel by incorporating visual attention mechanisms. The goal of these techniques is to explore, in detail, the objects or regions that stand out in a given scene [47], as opposed to convolutional methods, whose kernels treat equally the whole content in the image. The main idea is to simulate the human behavior, as we try to understand an image by selecting a subset of features that contain the most relevant characteristics instead of treating the full scene equally. In fact, the human brain focuses on the most valuable and informative stimulus perceived by the eyes, ignoring other irrelevant information. Such visual attention mechanisms are based on two kinds of components [48]: 1) *bottom-up* (stimulus-driven) components that are traditionally related with automatic/involuntary processing of salient visual features in raw sensory information and are performed in a feedforward way and 2) *top-down* (goal-oriented) components that modulate bottom-up component behavior through voluntary attention to certain characteristics, objects, or regions in the space. The study of these components, together with their characteristics, has resulted in a great variety of attention-driven techniques [49], turning visual attention into a hot research topic.

In the remote sensing literature, several attention-driven techniques have been developed for detecting salient regions [50]–[56] and target objects [57]–[60]. However, their application to HSI data has been quite sparse [61], [62]. Although the adaptation of visual attention techniques to deep models is demonstrating excellent performance in several classification tasks [63]–[65], there is still room for contributions in the area of HSI classification.

In this paper, we develop a new spectral–spatial visual attention-driven technique for HSI classification. Our newly developed technique combines the use of advanced visual attention mechanisms with powerful feature extraction approaches based on DNNs for spectral–spatial HSI classification. As a case study, we introduce visual attention mechanisms in the ResNet architecture (A-ResNet). The translation of a visual attention working mode to DNNs for HSI data processing allows to increase the sensitivity of the network to those features that contain the most important and useful information for classification purposes. In this regard, the main innovative contributions of our work can be summarized as follows.

1) The development, for the first time in the literature, of a visual attention-driven mechanism (incorporated into an A-ResNet) for spatial–spectral HSI classification. This is done by introducing a dual data-path attentional module as the basic building module, considering both bottom-up and top-down visual factors to improve the feature extraction capability of the network.

2) A detailed comparison between our attention-driven model and traditional pixel-based machine learning and spectral–spatial DL-based techniques for HSI classification, demonstrating that the proposed model is able to outperform the current state-of-the-art classifiers.
3) A study of how the performance of the considered classifiers is affected by perturbations in the data, introducing controlled noise in the samples. To this end, four well-known and publicly available HSIs are considered in our experiments: Indian Pines (IP), University of Pavia (UP), Salinas Valley (SV), and University of Houston (UH).

The remainder of this paper is organized as follows. In section II, we introduce the basic principles of CNNs and the ResNet model. Section III describes, in detail, our newly proposed A-ResNet methodology. Section IV discusses our experimental results. Finally, Section V concludes this paper with some remarks and hints at plausible future research lines.

II. Related Work

A. Convolutional Neural Networks

DNNs are characterized by a hierarchical structure composed by a deep stack of processing layers, placed one after the other. Such deep structure allows these models to learn representations of the original input data with multiple levels of abstraction, from the most concise ones (at the first layers) to the most abstract ones (at the end of the architecture). Such multilevel representations of the data allow for a powerful mechanism of feature extraction, in which each layer is able to discover (or reinforce) different relations, distributions, and structures in the data, supported by features extracted by previous layers. In this sense, the architecture of CNNs is based on receptive fields and follows the behavior of neurons in the primary visual cortex of a biological brain [66], [67]. These models have become a state of the art in remote sensing data analysis, outperforming many algorithms [68]. CNNs are typically composed of two main parts: 1) the feature extractor net, and 2) the classifier.

The feature extractor is composed by several kinds of n-dimensional blocks or layers, depending on how the information is used and how it is processed by these blocks. An HSI data set $\mathbf{X}$ can be seen as a collection of spectral vectors $\mathbf{X} \in \mathbb{R}^{(n_1 \cdot n_2) \times n_{\text{bands}}}$, where $n_1 \cdot n_2$ denotes the number of spectral pixels in the scene and n_{bands} is the number of spectral bands. Each pixel in the scene is given by $\mathbf{x}_i \in \mathbb{R}^{n_{\text{bands}}} = [x_{i,1}, x_{i,2}, \ldots x_{i,n_{\text{bands}}}]$. CNN models composed by 1-D blocks process only the spectral information in the data and are also known as spectral-based CNNs. These models exhibit similar disadvantages as traditional pixel-based processing methods. On the contrary, if we apply a spectral dimension reduction technique over $\mathbf{X}$, for example, PCA [69], [70], and retain only the first PC, the HSI can be treated as a 2-D matrix of spatial information $\mathbf{X} \in \mathbb{R}^{n_1 \times n_2}$, where $n_1 \times n_2$ denotes the number of rows and columns in the scene. Traditional CNNs employ 2-D blocks to process the spatial information contained in the input data, which, in RGB data, corresponds with the whole image. However, to process the HSI $\mathbf{X}$ using both spatial and spectral information, we need to extract, for each pixel $\mathbf{x}_{i,j} \in \mathbb{R}^{n_{\text{bands}}}$, a neighborhood window or spatial patch $\mathbf{p}_{i,j} \in \mathbb{R}^{d \times d}$, which comprises the set of $d \times d$ pixels that surround the central sample $\mathbf{x}_{i,j}$. The usual way to perform the classification is to assign the label $y_{i,j}$ of the central pixel $\mathbf{x}_{i,j}$ to the entire patch $\mathbf{p}_{i,j}$. Although such a spatial-based classification strategy can achieve good results, the loss of significant spectral information is often critical in many applications [37], [38]. A third way to classify the HSI scene $\mathbf{X}$ is to exploit the spatial-contextual information together with the full or filtered spectra, retaining the full spectral information from the original bands (or a significant percentage of it, by means of an appropriate number of principal components) and creating spectral–spatial patches or data subcubes $\mathbf{p}_{i,j} \in \mathbb{R}^{d \times d \times n_{\text{channels}}}$. In this sense, the spectral–spatial CNN model allows to treat the data in 3-D fashion by combining both sources of information (spatial and spectral) in a most natural and simple way, by considering 3-D subblocks extracted from the input data cube.

Using spectral–spatial patches as inputs, the feature extractor net of the spectral–spatial CNN model hierarchically applies three kinds of operations: 1) *convolution*; 2) nonlinear *activation*; and 3) donwsampling by *pooling*. The convolutional layer is the main processing block, composed by K filters defined by their receptive field. In this sense, regarding the dimension of the filters, the CNN can be understood as 1-D, 2-D, or 3-D depending on whether its receptive field is of dimensions $K \times q$, $K \times k \times k$, or $K \times k \times k \times q$, respectively, being q and k the spectral and spatial components of the kernel (in this context, the proposed model implements a spectral–spatial convolutional-based model with 2-D kernels). In fact, the convolutional layer can be interpreted as a sliding-window method, where the windows/kernels of the block slide over the spatial and spectral dimensions of the input volume using a stride $s^{(l)}$

$$\mathbf{X}^{(l)} = \mathbf{W}^{(l)} * \mathbf{X}^{(l-1)} + \mathbf{b}^{(l)} \tag{1}$$

where $\mathbf{X}^{(l)}$ is the output volume of the lth layer, composed by K feature maps and obtained as the convolution ($*$) of the input volume $\mathbf{X}^{(l-1)}$ and the layer weights $\mathbf{W}^{(l)}$ and biases $\mathbf{b}^{(l)}$. More specifically, each feature of $\mathbf{X}^{(l)}$ in (1) is obtained as follows:

$$\begin{aligned} x_{i,j}^{(l)z} &= (\mathbf{W}^{(l)} * \mathbf{X}^{(l-1)} + \mathbf{b}^{(l)})_{i,j} \\ &= \sum_{\hat{i}=0}^{k^{(l)}-1} \sum_{\hat{j}=0}^{k^{(l)}-1} \left(\mathbf{x}^{(l-1)}_{(i \cdot s^{(l)}+\hat{i}),(j \cdot s^{(l)}+\hat{j})} \cdot \mathbf{w}^{(l)}_{\hat{i},\hat{j}} \right) + \mathbf{b}^{(l)} \end{aligned} \tag{2}$$

where $x_{i,j}^{(l)z} \in \mathbb{R}$ is the (i, j)th element of the zth feature map of $\mathbf{X}^{(l)}$ (with $z = 0, 1, \ldots, K^{(l)} - 1$ and $K^{(l)}$ being the number of filters of the layer), $\mathbf{x}_{i,j}^{(l-1)} \in \mathbb{R}^{K^{(l-1)}}$ is the (i, j)th element of the input volume $\mathbf{X}^{(l-1)}$, $\mathbf{w}^{(l)}_{\hat{i},\hat{j}}$ is the $(\hat{i}, \hat{j})$th weight of the layer weights $\mathbf{W}^{(l)}$, $\mathbf{b}^{(l)}$ denotes the biases, and $s^{(l)}$ is the stride, being $k^{(l)} \times k^{(l)}$ the receptive field of the lth layer. Fig. 1 presents a graphical visualization of the operations conducted by (1) and (2).

Convolutional blocks extract the features contained in the input volume by applying a linear dot product. In order to learn nonlinear relationships present in the data, a nonlinear

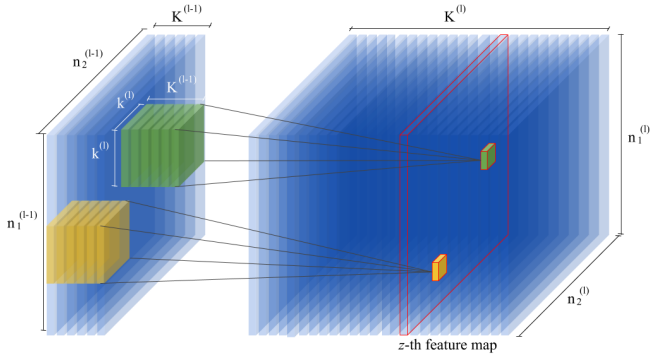


Fig. 1. Visualization of a convolutional layer operation with 2-D kernel. Unlike fully connected layers, the lth convolutional block presents local connectivity to small regions of the whole input volume, that is, the zth filter's weights $\mathbf{W}^{(l)}$ are applied over windows of the input volume $\mathbf{X}^{(l-1)} \in \mathbb{R}^{n_1^{(l-1)} \times n_2^{(l-1)} \times K^{(l-1)}}$ defined by the receptive field of size $k^{(l)} \times k^{(l)}$, taking into account the full depth $K^{(l-1)}$ of the input data (highlighted as green and yellow patches), slipped by a stride determined by $s^{(l)}$. It can be observed that the zth kernel produces, for each region, a scalar value (represented as a smaller rectangle) that is allocated into the zth feature map, giving, as a result, an output volume $\mathbf{X}^{(l)} \in \mathbb{R}^{n_1^{(l)} \times n_2^{(l)} \times K^{(l)}}$ that comprises $K^{(l)}$ feature maps of $n_1^{(l)} \times n_2^{(l)}$ features each.

activation function is adopted before sending the resulting output volume to the following layer $\mathbf{X}^{(l)} = \mathcal{H}(\mathbf{X}^{(l)})$, being $\mathcal{H}(\cdot)$ usually implemented by the rectified linear unit (ReLU) [71]. In addition, with the aim of reducing the spatial dimensions of the output volume and also to summarize the obtained features and obtain a certain invariability to geometric transformations, a nonlinear subsampling strategy is implemented by the pooling layer. In fact, the pooling layer applies a sample-based discretization process, selecting from small windows of the input volume those values that satisfy the selection criteria, being the max-pooling one of the most widely used methods for this purpose. It simply slides a spatial kernel $k \times k$ over the input volume, selecting the maximum value for each region, as the following equation indicates:

$$\text{pool}_{i,j}^{(l)_z} = \max_{(a,b) \in \mathcal{R}_{i,j}} x_{a,b}^{(l)_z} \tag{3}$$

where $\text{pool}_{i,j}^{(l)_z}$ represents the (i, j)th output value of the pooling associated with the zth feature map and $x_{a,b}^{(l)_z}$ denotes the (a, b)th element contained by the pooling region $\mathcal{R}_{i,j}$ that encapsulates a spatial receptive field around the position (i, j) [72].

At the end of the feature extractor net, a final output $\mathbf{X}^{(l)}$ is obtained that contains an abstract representation of the original input data. Usually, this output is flattened in order to allow the classifier to perform the final categorization of the input data. Normally, the classifier is implemented by one or more fully connected layers of a multilayer perceptron (MLP), creating an end-to-end structure.

B. Residual Neural Networks

CNNs present several problems when processing HSI data. In particular, they tend to overfit when very few labeled samples are available to perform the training procedure, and

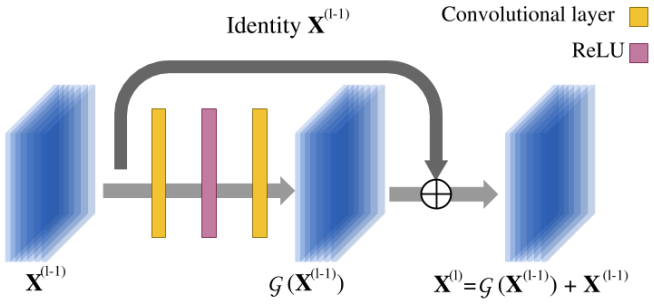


Fig. 2. Graphic visualization of a standard residual unit. The final output volume is obtained as the aggregation of the original input volume $\mathbf{X}^{(l-1)}$ and the resulting output volume of the hidden stack of layers, $\mathcal{G}(\mathbf{X}^{(l-1)})$, where $\mathcal{G}(\cdot)$ refers to the convolutions, normalizations, pooling steps, and activation functions applied along the stack over the input data. As a result, the architecture reinforces the learning process of the entire model by reusing previous information in the following layers: $\mathbf{X}^{(l)} = \mathcal{G}(\mathbf{X}^{(l-1)}) + \mathbf{X}^{(l-1)}$.

they also can suffer from loss of information when deep structures are implemented. To overcome the first problem, several strategies have been developed in the literature, such as the use of data regularization and dropout techniques, data augmenting, or semisupervised and AL approaches. However, the loss of information is produced by the vanishing gradient problem [73]. In this case, for very deep architectures, the errors become quite hard to propagate back correctly, and the gradient signal tends to zero [74]. Several strategies have also been developed to deal with this problem, such as data normalization techniques [75] or new optimizer/activation functions [76], [77]. However, the accuracy of deep CNNs still can saturate due to the complexity of the mapping function of the convolutional blocks and the hard learning of these functions [78]. In this sense, the architectural modifications introduced by ResNets can improve the learning process of convolutional layers by learning small residuals and adding them to the input volume of each layer, instead of transforming the whole input volume directly. In order to differentiate the CNN and ResNet models, we note that the main building block of a CNN is composed by the convolutional layer and the nonlinear activation function, so (1) with $\mathcal{H}(\cdot)$ can be rewritten as

$$\begin{aligned} \mathbf{X}^{(l)} &= \mathcal{H}(\mathbf{W}^{(l)} * \mathbf{X}^{(l-1)} + \mathbf{b}^{(l)}) \\ \text{simplifying } \mathbf{X}^{(l)} &= \mathcal{H}(\mathbf{X}^{(l-1)}) \end{aligned} \tag{4}$$

Equation (4) indicates that the CNN hierarchically extracts the features, processing them by the successive layers that compose the architecture. Instead of that, the ResNet uses the residual unit as a building block [79] and is composed by a stack of several layers, normally convolutional layers stacked with ReLUs and batch-normalization layers, and with two types of connections allowing different kinds of data streams (see Fig. 2).

1) The traditional forward connection that connects the current layer with the previous and the following ones, extracting from the original input volume $\mathbf{X}^{(l-1)}$ a representation $\mathcal{G}(\mathbf{X}^{(l-1)}, \mathcal{W}^{(l)}, \mathcal{B}^{(l)})$, where $\mathcal{G}(\cdot)$ approximates the residual function referring to those operations that

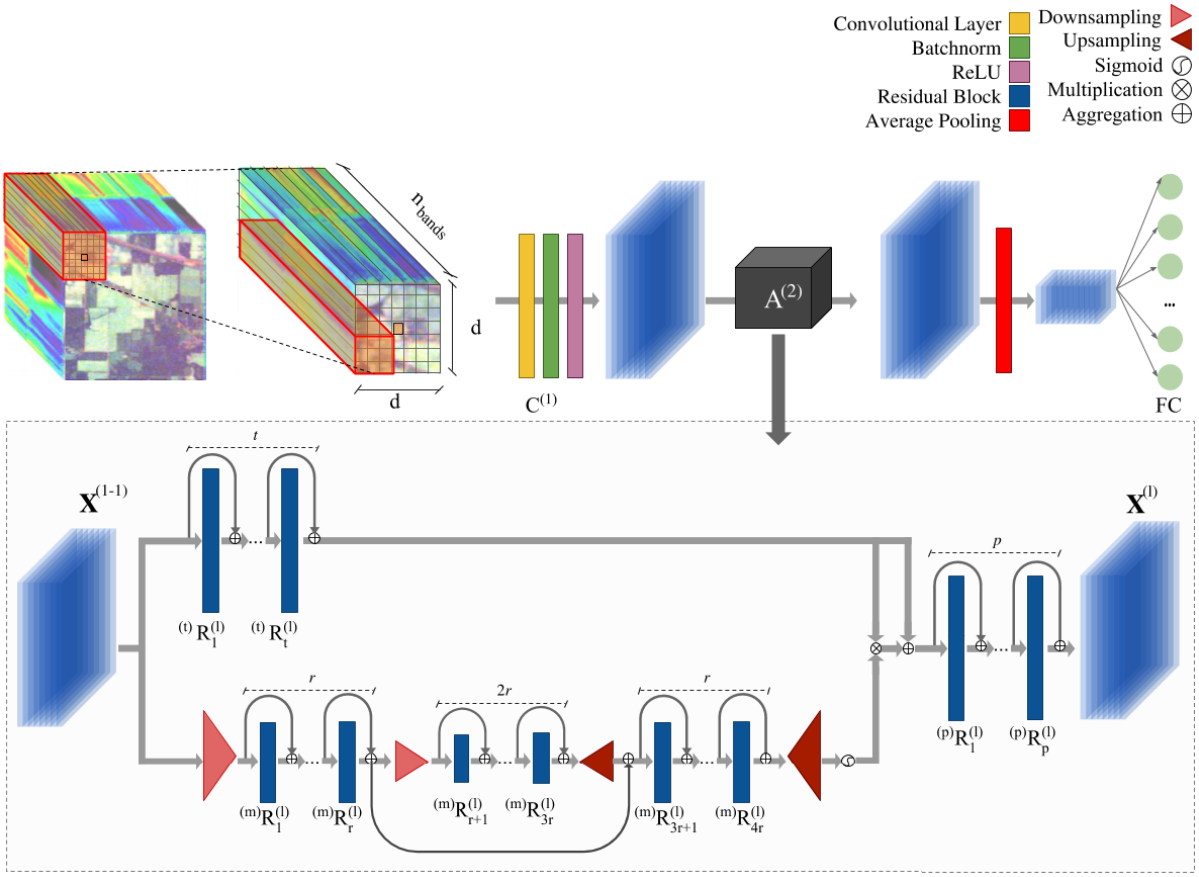


Fig. 3. Standard architecture of the proposed network with the network's head, composed by a convolutional layer $C^{(1)}$ that presents the input volume data $\mathbf{X}$, to the network's body, composed by the residual attention module, $A^{(2)}$, whose output is finally vectored through an average pooling and sent to the network's tail, composed by one fully connected layer that performs the final classification. Two branches, trunk and mask, compose the attentional module: the trunk branch (upper path), composed by t residual blocks that perform feature extraction from the data, and the mask branch (bottom path), composed by a symmetrical downsampler–upsampler structure, in which r residual blocks are allocated (in between each downsampling/upsampling step) to extract information from the current scale, adding a shortcut connection to link the downsampling step ($/2$) with its corresponding upsampling ($\times 2$) counterpart to combine both kinds of data (instead of the bottleneck part, where only $2 \cdot r$ residual blocks are stacked one after the other), and followed by a sigmoid function to prepare the mask, which is applied over the trunk feature data. The resulting output is sent to a final group of p residual blocks located at the end of the module.

are applied over the input data by all the stacked layers of the residual unit, which depends on the weight matrices $\mathcal{W}^{(l)} = \{\mathbf{W}^{(i)}\}_{i=0}^{N-1}$ of the N convolutional layers associated with the lth residual unit, and the corresponding biases $\mathcal{B}^{(l)} = \{\mathbf{b}^{(i)}\}_{i=0}^{N-1}$.

2) The shortcut connection that communicates the original input volume with the end of the residual unit, performing an identity mapping that allows to reuse the previous information to reinforce the learning of the residual block.

At the end, residual learning is introduced into (1) as

$$\mathbf{X}^{(l)} = \mathcal{G}(\mathbf{X}^{(l-1)}, \mathcal{W}^{(l)}, \mathcal{B}^{(l)}) + \mathbf{X}^{(l-1)}$$

$$\text{simplifying: } \mathbf{X}^{(l)} = \mathcal{G}(\mathbf{X}^{(l-1)}) + \mathbf{X}^{(l-1)} \tag{5}$$

where the previous features are exploited once again by the next unit, which reinforces the learning and allows the gradient to be transmitted.

III. Attentional Residual Network for Hyperspectral Image Classification

The combination of convolutional kernels and residual connections makes the ResNet a very powerful and efficient model for image analysis, in general, and for HSI processing, in particular. Based on this architecture, this section develops a new architecture for HSI classification that incorporates visual attention mechanisms in order to extract more discriminatory features, improving the model performance and enhancing its accuracy. In this sense, analogous to the original ResNet, the proposed spectral–spatial A-ResNet for HSI classification adopts a basic building block, called attentional module [65], that contains two data paths or branches: 1) the *trunk branch* and 2) the *mask branch*. Fig. 3 presents the overall architecture of the proposed attentional neural network for HSI data classification. Focusing on the attentional module, the specifications of each part are discussed in detail in the following.

A. Attentional Module → Trunk Branch

The attentional module can be denoted as $A^{(l)}$, with l being the number of layers, and receives the volume $\mathbf{X}^{(l-1)}$ as input data, which is forward-propagated through two different paths, being the trunk branch the simplest and easiest one to implement. It is composed by t residual blocks, which are stacked one by one, performing a feature extraction and processing task. These residual blocks can be implemented following previous works, such as the basic residual block and its bottleneck implementation [78], the wide residual block [80], and the pyramidal residual block and its bottleneck variation [41], [81], among other complex structures [79], [82], [83]. The obtained features can be denoted as $\mathbf{X}^{(l_{\text{trunk}})} = \text{trunk}(\mathbf{X}^{(l-1)})$ and contain the high-level data representation of the module. At this point, and following visual attention principles, the next step is to single out the most relevant features from all of the available information contained into $\mathbf{X}^{(l_{\text{trunk}})}$, masking the least interesting parts for the learning procedure. In this sense, an attention mask $\mathbf{X}^{(l_{\text{mask}})}$ must be calculated and applied over the processed features of the trunk branch.

B. Attentional Module → Mask Branch

As mentioned earlier, the input module $\mathbf{X}^{(l-1)}$ is propagated through two paths, with the mask branch being in charge of calculating and applying the attention mask $\mathbf{X}^{(l_{\text{mask}})}$ over the output features obtained by the trunk branch, $\mathbf{X}^{(l_{\text{trunk}})}$. In fact, its goal is to obtain a weight matrix with the same dimensions of $\mathbf{X}^{(l_{\text{trunk}})}$, which softly weights the trunk's output features to highlight the most important ones, simulating the elementwise soft attention mechanism.

In order to obtain the final $\mathbf{X}^{(l_{\text{mask}})}$, the mask branch applies a network architecture over $\mathbf{X}^{(l-1)}$. It is based on a spatial downsampler–upsampler structure with r residual blocks, allocated between each pair of downsampling/upsampling steps and with skip connections between each downsampling step and its upsampling counterpart (similar to the hourglass network [84]), following the anatomical connections of cortical processing [85], where feedforward connections transform the input into fast behavioral responses, whereas skip/feedback connections modulate these responses using perceptual context or attention. Moreover, each sampling step (coupled with its corresponding r residual blocks) provides semantic information about the input data, from low-level cues (edges, color, and intensity) to high-level cues that, coupled with the forward connections (aimed at collecting global information from the data) and skip connections (which allow to combine multiscale data taking into account global information and original features) simulate the bottom-up and the top-down attention selections of the visual cortex [86]. In this sense, the downsampler–upsampler structure stacks as many downsampling/upsampling steps as possible, until the smallest feasible spatial resolution of the data is reached.

In the attention module $A^{(l)}$, the naive application of the attentional mask over the trunk features in the spatial–spectral domain gives the following output:

$$\mathbf{X}^{(l)} = \mathbf{X}^{(l_{\text{mask}})} \cdot \mathbf{X}^{(l_{\text{trunk}})}. \tag{6}$$

However, (6) presents several limitations. Considering the mask $\mathbf{X}^{(l_{\text{mask}})}$ as a collection of values in the range [0, 1], its application over trunk features may degrade them in deeper layers. Also, if the mask contains in most of its elements a value that is equal or close to 0, it may disregard relevant features of the trunk branch. In order to overcome these problems, (6) is reformulated as follows:

$$\mathbf{X}^{(l)} = (1 + \mathbf{X}^{(l_{\text{mask}})}) \cdot \mathbf{X}^{(l_{\text{trunk}})}. \tag{7}$$

In this case, (7) allows propagating the characteristics extracted from the trunk branch, where the mask branch suppresses the least significant features to facilitate the detection of important features. The combination of both allows to single out the salient features.

Finally, the masked output volume is passed through a tail composed by p residual blocks that perform a final feature extraction step, taking into account the features that have been highlighted in the previous phase.

C. Proposed Network Topology

The proposed network for spectral–spatial HSI data classification has been developed to work with 3-D subcubes $\mathbf{p}_{i,j}\,\mathbb{R}^{d\times d\times n_{\text{channels}}}$ extracted around each spectral pixel $\mathbf{x}_{i,j}$ of the original scene, taking $d = 11$ as the spatial height and width dimensions [40]. These input patches are passed through the network, which is composed by the network's head, attentional body, and classification tail (see Fig. 3) in order to extract relevant features and perform their corresponding classification. The head of the network is given by a convolutional layer $C^{(1)}$ with batch-normalization and ReLU, which prepares the data to be processed by the rest of the network, followed by one or several attentional modules, depending on the complexity of the problem. As mentioned earlier, the lth attentional module $A^{(l)}$ is, in turn, composed by several residual blocks ${}^{*}R_i^{(l)}$ (see Fig. 3):

1) t residual blocks, denoted as ${}^{(t)}R_i^{(l)}$, with $i = 1, \ldots, t$, for extracting features in the trunk branch;
2) r(2DU) residual blocks, denoted as ${}^{(m)}R_i^{(l)}$, being DU the number of down sampling/upsampling steps for processing multiscale data and obtain the attention module mask. For instance, in Fig. 3, with DU = 2 downsampling/upsampling steps, there are $4r$ residual blocks
3) p residual blocks denoted as ${}^{p}R_i^{(l)}$ with $i = 1, \ldots, p$, located at the end of the module for postprocessing the filtered data.

In total, the attention module is composed by $t+r$(2DU)$+p$ residual blocks, being $t = 2$, $r = 1$, and $p = 1$, while DU depends on the spatial size of the input volume. The residual block architecture of the trunk branch is composed by three subblocks of convolutional layers, batch-normalization, and ReLU (see Fig. 4), whose kernels are defined in Table I, creating a spectral-bottleneck architecture in order to better analyze the spectral–spatial domains [87], while the residual blocks of the mask and the ending of the module follow the simple residual unit designed in [78]. Kernels are defined in Table I. As we can observe, each kernel performs a

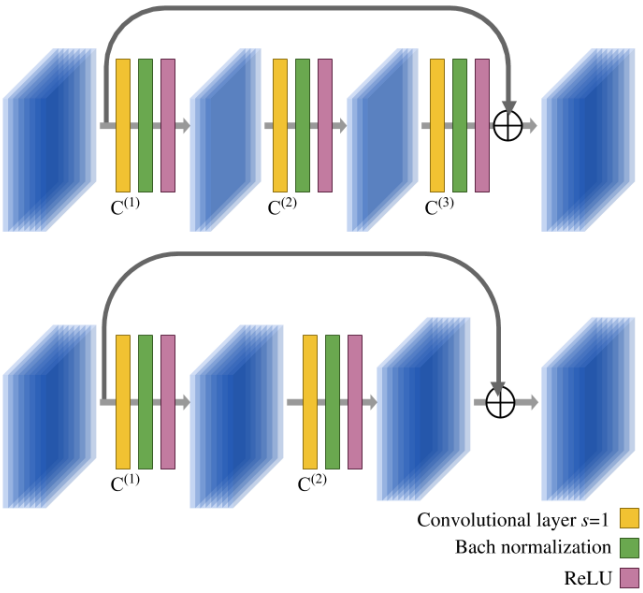


Fig. 4. (Top) Graphic visualization of the architecture of the internal residual blocks that conform the trunk branch of the attentional module and (Bottom) those that conform the mask branch. Convolutional details are given in Table I.

TABLE I

BASIC ARCHITECTURE OF THE RESIDUAL BLOCKS OF THE TRUNK AND MASK BRANCHES, WHERE $K_{\text{MIDDLE}} = K_{\text{INPUT}}/2$

Layer ID	Kernel size	Stride	Padding
Bottleneck residual block from trunk branch			
$C^{(1)}$	$K_{middle} \times 3 \times 3 \times K_{input}$	$s = 1$	$p = 1$
$C^{(2)}$	$K_{middle} \times 3 \times 3 \times K_{middle}$	$s = 1$	$p = 1$
$C^{(3)}$	$K_{input} \times 3 \times 3 \times K_{middle}$	$s = 1$	$p = 1$
Residual blocks from trunk branch			
$C^{(1)}$	$K_{input} \times 3 \times 3 \times K_{input}$	$s = 1$	$p = 1$
$C^{(2)}$	$K_{input} \times 3 \times 3 \times K_{input}$	$s = 1$	$p = 1$

convolution operation using windows of size 3×3, with padding $p = 1$. In this context, the output of the attention module, $\mathbf{X}^{(l)}$, maintains the same spatial–spectral dimensions as the input, $\mathbf{X}^{(l-1)}$, in the sense that all its residual blocks keep the volume dimensions constant. This allows us to add a lot of flexibility to the model, which is able to stack modules one after another (as *plug-&-play* structures). In order to avoid the overfitting problem caused by a large number of parameters that must be trained, we propose a simple architecture with one attentional module. Details can be found on Table II.

Furthermore, the network has been optimized using the Adam optimizer [76] with 300 epochs, where the learning rate decays half of its value on epochs 50, 100, and 200, using a batch size of 100. Also, $n_{\text{channels}} = 40$ principal components have been considered as the input spectral bands, being $d = 11$.

IV. EXPERIMENTAL RESULTS

A. Experimental Configuration

With the aim of testing the performance of the proposed attentional network for spectral–spatial HSI classification, a battery of experiments has been performed on a desktop computer equipped with a sixth-generation Intel Core i7-6700K processor, with 8M of cache, the clock speed of 4.20 GHz, and four cores/eight-way multitask processing. From the point of view of memory, it is equipped with 40 GB of DDR4 RAM, with a serial speed of 2400 MHz, and a Toshiba DT01ACA HDD with 7200 rpm and 2 TB of storage capacity. Also, it is equipped with a graphics processing unit (GPU) NVIDIA GeForce GTX 1080 with 8-GB GDDR5X of video memory and 10 Gb/s of memory frequency, and an ASUS Z170 programming motherboard. The operating system is Ubuntu 18.04. In order to efficiently implement the proposed approach, our models have been parallelized on the available GPU using Pytorch.

TABLE II

TOPOLOGY OF THE PROPOSED ATTENTION NETWORK, WHERE n_{CHANNELS} INDICATES THE NUMBER OF CONSIDERED SPECTRAL BANDS

Input convolutional layer		
ID	**Kernel size**	**Stride**
$C^{(1)}$	$64 \times 1 \times 1 \times n_{channels}$	$s = 1$
Attention module		
ID	**Processed data**	**Parameters**
$A^{(2)}$	$11 \times 11 \times 64$	$t = 2$ $r = 1$ $p = 1$ $DU = 2$
Average pool		
ID	**Kernel**	
$AVPOOL$	2×2	
Fully connected layer		
ID	**Input $\times$ output neurons**	**Activation**
FC	$576 \times n_{classes}$	Softmax

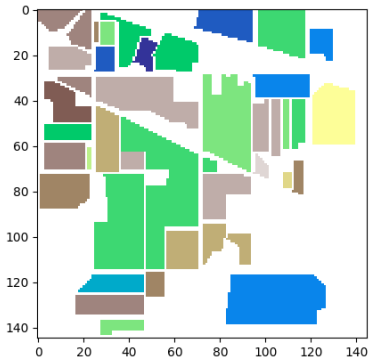
B. Hyperspectral Data Sets

Four public and widely used HSI data sets have been considered in our experiments: IP, UP, SV, and the Kennedy Space Center (KSC). Table III shows, for each data set, its corresponding ground-truth with the number of samples per class. In the following, we summarize the characteristics of each data set.

1) IP data set was collected by the Airborne Visible InfraRed Imaging Spectrometer (AVIRIS) [88] in 1992, over an agricultural area in Northwestern Indiana using 145×145 pixels with a spatial resolution of 20 meters/pixel (m/p) and 224 spectral bands in the wavelength range from 0.4 to 2.5 μm. After deleting 24 bands due to water absorption and null values, a total of 200 spectral bands are considered for experimental purposes. The ground-truth is divided into 16 different classes (see Table III).
2) UP data set was collected by the reflective optics system imaging spectrometer (ROSIS) [89] in 2002, over the Engineering School at the UP, Northern Italy, using 610×340 pixels with a spatial resolution of 1.3 m/p and 103 spectral bands in the wavelength range from 0.43 to 0.86 μm. The ground-truth is divided into nine different classes (see Table III).

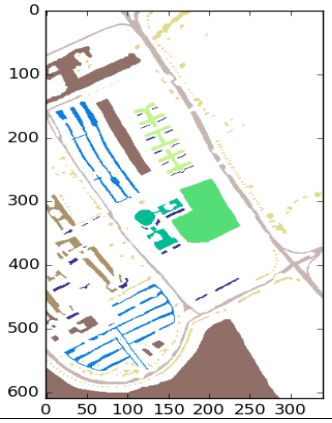
TABLE III
NUMBER OF SAMPLES OF THE IP, UP, SV, AND UH DATA SETS

INDIAN PINES (IP)



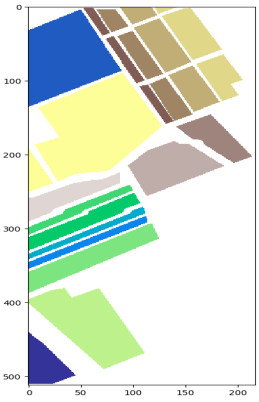
Color	Land-cover type	Samples
	Background	10776
	Alfalfa	46
	Corn-notill	1428
	Corn-min	830
	Corn	237
	Grass/Pasture	483
	Grass/Trees	730
	Grass/pasture-mowed	28
	Hay-windrowed	478
	Oats	20
	Soybeans-notill	972
	Soybeans-min	2455
	Soybean-clean	593
	Wheat	205
	Woods	1265
	Bldg-Grass-Tree-Drives	386
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

UNIVERSITY OF PAVIA (UP)



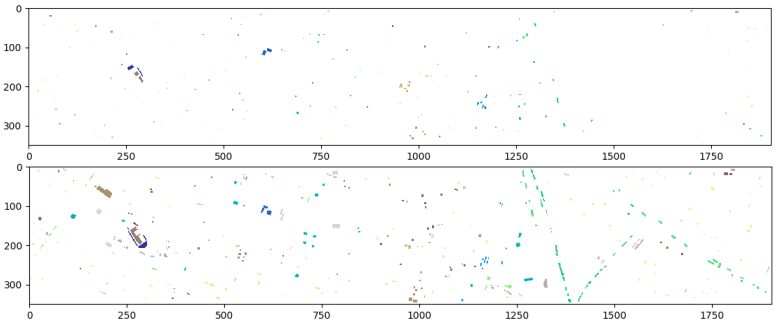
Color	Land-cover type	Samples
	Background	164624
	Asphalt	6631
	Meadows	18649
	Gravel	2099
	Trees	3064
	Painted metal sheets	1345
	Bare Soil	5029
	Bitumen	1330
	Self-Blocking Bricks	3682
	Shadows	947
	Total samples	207400

SALINAS (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

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

3) SV data set was collected by the AVIRIS sensor in 1998, over an agricultural field in Salinas Valley, CA, USA, using 512 × 217 spectral samples with 224 spectral bands (20 of them were discarded due to water absorption and noise). The ground-truth contains 16 classes (see Table III).

4) UH data set [90] provides an interesting benchmark, first presented by the IEEE Geoscience and Remote Sensing Society Image Analysis and the Data Fusion Technical Committee during the 2013 data fusion contest [91]. It was gathered by the Compact Airborne Spectrographic Imager (CASI) in June 2012, over the campus of the

University of Houston and the neighboring urban area, forming a data cube of dimensions $349 \times 1905 \times 144$, with a spatial resolution of 2.5 m and spectral information captured in the range from 0.38 to 1.05 μm, containing 15 ground-truth classes divided in two categories: training (top UH map in Table III) and testing (bottom UH map in Table III).

C. Results and Discussion

In order to test the performance of proposed attention-guided network for spectral–spatial HSI data classification, four main experiments have been carried out.

1) Our first experiment performs a comparison between the proposed attention-driven network and seven different and widely used HSI classifiers available in the literature: 1) RF; 2) multinomial LR (MLR); 3) SVM; 4) MLP; 5) spectral CNN (CNN1D); 6) spatial CNN (CNN2D); and 7) spectral–spatial ResNet. In this context, the four HSI data sets described in Section IV-B have been used. We extracted patches of size $11 \times 11 \times 40$. For the IP scene, we used 15% of the available labeled data per class for training (and the rest of the available labeled data for testing). For the UP and SV scenes, we used 10% of the available labeled data for training. Finally, for the UH scene, we used the available (fixed) training set (see Table III).
2) Our second experiment expands the initial comparison carried out in the first experiment using different classifiers and particularly focusing on different spectral–spatial methods carried out on the UP data set with the fixed training set adopted in [92]. In this case, the following classifiers have been considered: 1) Markov random field combined with the Gaussian class-conditional model (MRF-Gauss); 2) contextual SVM (CSVM) [93]; 3) CNN with extinction profiles (EP-CNN) [94]; 4) CNN with a previously applied PCA; 5) CNN with extended morphological profiles (EMP-CNN); and 6) CNN with Gabor filter (Gabor-CNN). Focusing on convolutional models, the EP-CNN is fed by patches of size $27 \times 27 \times n_{\text{bands}}$, while the proposed attentional model, PCA-CNN, EMP-CNN, and Gabor-CNN employ the input patches of size $27 \times 27 \times 3$.
3) Our third experiment performs a comparison between the original spectral–spatial ResNet and the proposed A-ResNet, evaluating the evolution of the overall accuracy (OA) of both classifiers when different training ratios are considered for the IP, UP, and SV scenes. In particular, 5%, 10%, and 15% ratios have been considered for the IP scene, and 1%, 5%, and 10% ratios have been considered for the UP and SV scenes. Again, the input patches have been extracted with a size of $11 \times 11 \times 40$.
4) Finally, our fourth experiment analyzes, in detail, the performance of the proposed network as compared with the original ResNet model in the presence of noisy data. In this case, several levels of noise have been tested with noise being modeled as a normal distribution with $\mu = 0$ and $\sigma = \{0.10, 0.20, 0.40, 0.80, 1.60, 3.20, 6.40\}$.

In order to carry out the aforementioned comparisons, some widely used measures have been considered, including the OA and average accuracy (AA), the kappa coefficient (K), and the execution times (in seconds).

1) Experiment 1 (Comparison Between the Standard HSI Classifiers and the Proposed Methods): First experiment performs a comparison between the proposed network and some of the most well-known HSI classifiers available in the literature. These methods can be divided into spectral-based ones (RF, MLR, SVM, MLP, and CNN1D), spatial classifiers (CNN2D), and spectral–spatial classifiers (ResNet and A-ResNet). For all the spectral–spatial methods, the input patch size has been set to $11 \times 11 \times 40$. In order to perform a fair comparison, the ResNet has been implemented with the basic architecture of the proposed network in Table II, where the ResNet is composed by the same network's head and tail, and the same architecture of the trunk branch inside the network's body.

The obtained results are reported in Tables IV–VII, where the corresponding average and standard deviation values (obtained after five Monte Carlo runs) are also displayed. Focusing on the obtained OA values, we can observe that spatial and spectral–spatial methods are, in general, able to outperform pixel-based methods (RF, MLR, SVM, MLP, and CNN1D), being residual based models (i.e., ResNet and A-Resnet) able to outperform the results obtained by the CNN2D. Focusing on the ResNet and the proposed A-Resnet, the performance of the latter is better than the performance of the former, being able to reach higher OA values than the original ResNet, in particular, in the classification of the IP and SV scenes. Another interesting aspect is the AA, which is higher in the proposed A-ResNet than in the original ResNet, indicating that, on average, the high OA achieved is not due to peaks in, say, very well ranked classes, but to a generally better rank for all classes. This is also supported by the smaller standard deviation values exhibited by our A-ResNet. In particular, we can highlight the good performance of the proposed model in small classes, (for instance, Alfalfa and Oats in the IP scene or Lettuce romaine 6wk in the SV scene), where the A-ResNet is able to reach better accuracy values than the basic ResNet. Focusing on SV and UH scenes (in Tables VI and VII, respectively), the obtained OA values may lead us to think that both ResNet and A-ResNet exhibit similar behavior. However, the standard deviation of A-ResNet is significantly smaller, indicating more robust and stable results (as the AA scores also suggest).

In addition, some of the obtained classification maps are shown in Figs. 5–7. It can be observed that the classification maps obtained by pixel-based classifiers show salt-and-pepper noise in almost the full IP data set and in some classes of SV, particularly Vinyard-untrained and Grapes-untrained. In the UP scene, the RF missclassifies a large amount of pixels in the Bare Soil class, for instance. In contrast, spectral–spatial methods greatly reduce these effects, with ResNet and A-ResNet being able to obtain classification maps that are

TABLE IV
CLASSIFICATION RESULTS FOR IP DATA SET USING 15% OF THE AVAILABLE LABELED DATA

Class	RF	MLR	SVM	MLP	CNN1D	CNN2D	ResNet	A-ResNet
Alfalfa	20.00±8.01	32.82±13.51	62.05±12.07	50.77±9.65	44.61±5.28	75.38±10.20	84.62±3.62	**89.23**±1.92
Corn-notill	61.53±1.95	75.07±0.99	81.45±1.21	78.90±3.01	81.04±2.19	91.54±0.76	94.64±1.48	**97.69**±0.85
Corn-min	53.62±2.61	57.96±2.32	70.55±2.40	66.27±2.53	70.69±0.36	86.95±3.59	95.26±3.32	**99.29**±0.48
Corn	35.12±2.90	45.67±6.03	72.93±4.93	61.19±6.40	60.10±2.79	88.56±4.02	84.48±6.83	**92.24**±2.61
Grass/Pasture	84.39±4.29	86.98±1.94	93.17±2.26	89.61±2.65	92.34±0.88	86.05±2.40	96.49±0.55	**99.02**±0.74
Grass/Trees	96.10±0.93	96.36±0.94	97.32±0.26	96.55±0.39	97.29±1.24	96.13±1.92	98.06±0.97	**99.77**±0.38
Grass/pasture-mowed	29.57±10.79	47.83±15.80	84.35±3.48	75.65±5.90	69.57±11.99	82.61±11.00	85.22±5.22	**93.04**±5.90
Hay-windrowed	96.11±2.98	99.16±0.71	98.32±0.66	97.54±1.36	98.18±0.79	97.88±0.40	100.00±0.00	100.00±0.00
Oats	1.18±2.35	18.82±6.86	51.76±12.56	61.18±15.16	44.70±16.89	65.88±21.18	68.24±10.91	**90.59**±7.98
Soybeans-notill	65.96±2.63	66.54±1.77	77.87±2.13	78.18±5.23	78.67±1.92	89.85±2.91	94.65±1.65	**98.57**±0.51
Soybeans-min	89.13±3.07	79.53±1.71	85.10±0.72	86.10±2.71	83.42±3.44	95.28±1.45	97.57±0.77	**99.37**±0.18
Soybean-clean	46.59±4.62	58.25±3.33	79.09±0.99	78.85±3.36	83.97±1.05	88.65±2.04	90.28±3.77	**97.14**±0.87
Wheat	92.18±4.20	98.51±0.59	98.39±1.23	98.74±0.67	98.62±0.28	97.82±2.42	99.89±0.23	**100.00**±0.00
Woods	94.53±0.59	95.31±0.75	95.59±0.54	94.55±1.30	94.51±0.97	98.40±0.57	99.14±0.32	**99.57**±0.31
Bldg-Grass-Tree-Drives	40.55±5.01	63.90±2.81	61.28±3.42	65.55±3.48	67.44±4.86	89.21±6.02	93.54±3.53	**99.58**±0.41
Stone-steel towers	83.54±1.96	85.06±2.58	87.60±5.74	89.37±4.43	87.59±3.53	82.53±6.27	89.87±6.46	**97.72**±1.68
OA	75.31±0.48	77.76±0.48	84.48±0.23	83.50±0.47	84.02±0.83	92.69±0.53	95.94±1.32	**98.75**±0.31
AA	61.88±0.98	69.24±1.51	81.05±1.44	79.31±1.23	78.30±1.01	88.29±2.01	92.00±2.27	**97.05**±1.01
K(x100)	71.41±0.54	74.46±0.56	82.26±0.28	81.13±0.54	81.75±0.90	91.65±0.60	95.37±1.51	**98.58**±0.36
Time (s.)	1.29±0.54	6.05±0.56	**0.25**±0.28	26.46±0.54	53.91±0.90	59.28±0.60	61.57±1.51	92.56±0.36

TABLE V
CLASSIFICATION RESULTS FOR UP DATA SET USING 10% OF THE AVAILABLE LABELED DATA

Class	RF	MLR	SVM	MLP	CNN1D	CNN2D	ResNet	A-ResNet
Asphalt	91.63±0.58	92.39±0.59	94.29±0.47	93.81±1.33	95.85±0.52	98.01±0.65	99.01±0.27	**99.80**±0.09
Meadows	97.71±0.36	96.09±0.48	97.49±0.12	97.58±0.45	98.13±0.41	99.41±0.15	99.91±0.03	**99.97**±0.03
Gravel	66.88±2.70	73.27±0.98	80.84±1.30	78.11±3.87	81.48±1.98	93.90±1.73	97.82±0.42	**99.56**±0.24
Trees	89.10±1.25	86.90±1.34	94.21±1.18	93.59±1.25	94.15±1.34	98.14±0.36	99.28±0.15	**99.74**±0.07
Painted metal sheets	98.60±0.39	99.59±0.31	99.22±0.31	99.52±0.16	99.82±0.08	99.57±0.35	99.92±0.13	**99.97**±0.04
Bare Soil	64.35±1.30	77.83±0.77	90.91±0.71	91.64±1.27	91.71±1.66	98.08±0.49	99.99±0.02	**100.00**±0.00
Bitumen	77.66±1.29	56.34±4.95	87.35±1.12	85.53±2.34	87.52±0.88	89.72±2.86	96.86±0.53	**99.16**±0.32
Self-Blocking Bricks	88.52±0.77	86.68±1.18	87.47±0.48	88.92±1.25	85.68±2.32	98.28±0.69	98.13±0.26	**99.73**±0.20
Shadows	99.74±0.23	99.67±0.12	99.86±0.09	99.53±0.25	99.88±0.07	98.87±0.51	**99.95**±0.06	99.88±0.10
OA	89.37±0.15	89.73±0.31	94.10±0.10	94.04±0.22	94.61±0.21	98.27±0.14	99.39±0.06	**99.86**±0.04
AA	86.02±0.29	85.41±0.63	92.40±0.16	92.02±0.45	92.69±0.10	97.11±0.25	98.99±0.12	**99.76**±0.05
K(x100)	85.67±0.20	86.27±0.41	92.17±0.14	92.09±0.28	92.84±0.28	97.71±0.18	99.19±0.09	**99.82**±0.05
Time (s.)	4.29±0.20	8.63±0.41	**0.44**±0.14	68.22±0.28	139.58±0.28	139.82±0.18	93.63±0.09	205.89±0.05

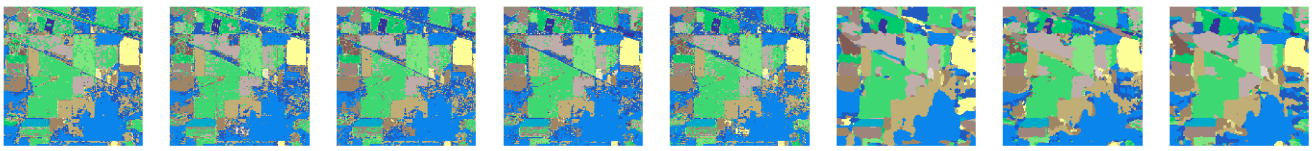
TABLE VI
CLASSIFICATION RESULTS FOR SV DATA SET USING 10% OF THE AVAILABLE LABELED DATA

Class	RF	MLR	SVM	MLP	CNN1D	CNN2D	ResNet	A-ResNet
Brocoli green weeds 1	99.46±0.14	99.47±0.16	99.63±0.20	99.57±0.12	99.88±0.10	99.45±0.32	99.61±0.47	**99.95**±0.04
Brocoli green weeds 2	99.83±0.05	99.94±0.06	99.91±0.08	99.87±0.09	99.96±0.02	99.51±0.38	99.99±0.01	99.99±0.01
Fallow	99.15±0.42	98.60±0.77	99.68±0.09	99.44±0.28	**99.85**±0.16	99.62±0.21	98.75±0.48	99.01±0.35
Fallow rough plow	99.42±0.25	99.28±0.29	99.31±0.33	99.25±0.58	99.57±0.15	99.89±0.16	99.76±0.20	**99.92**±0.07
Fallow smooth	97.87±0.38	99.12±0.32	99.35±0.17	99.09±0.42	99.05±0.47	**99.88**±0.10	99.25±0.61	99.86±0.19
Stubble	99.68±0.10	99.92±0.06	99.80±0.17	99.85±0.09	99.85±0.07	99.78±0.26	100.00±0.00	100.00±0.00
Celery	99.39±0.09	**99.89**±0.06	99.54±0.17	99.57±0.22	99.84±0.06	99.64±0.10	99.82±0.09	99.88±0.13
Grapes untrained	84.42±0.93	87.98±0.50	90.51±0.40	86.88±1.75	90.98±1.33	95.60±0.42	97.45±2.51	**99.77**±0.09
Soil vinyard develop	99.07±0.17	99.73±0.17	99.92±0.03	99.73±0.23	99.83±0.18	99.54±0.20	99.98±0.02	**99.99**±0.01
Corn senesced green weeds	91.56±1.09	95.79±0.54	97.71±0.48	96.56±1.05	98.03±0.22	98.45±0.84	99.38±0.39	**99.92**±0.07
Lettuce romaine 4wk	94.13±0.69	95.90±1.02	98.88±0.39	97.81±0.34	98.33±0.94	98.73±0.90	98.96±0.43	**99.60**±0.16
Lettuce romaine 5wk	98.79±0.23	99.63±0.15	99.79±0.07	99.65±0.12	99.96±0.03	99.58±0.51	100.00±0.00	100.00±0.00
Lettuce romaine 6wk	97.86±0.92	99.03±0.45	98.88±0.98	99.03±0.20	99.17±0.58	99.13±0.95	98.91±0.39	**99.76**±0.28
Lettuce romaine 7wk	91.34±1.77	96.03±0.70	97.65±1.34	96.80±0.82	97.34±0.80	97.53±0.84	99.48±0.52	**99.94**±0.05
Vinyard untrained	60.46±2.51	66.63±0.91	70.54±1.26	77.81±2.20	79.52±1.99	95.01±1.21	97.47±2.06	**99.84**±0.08
Vinyard vertical trellis	97.06±0.84	98.89±0.52	99.18±0.28	99.08±0.26	99.00±0.30	97.00±1.35	**99.91**±0.14	99.84±0.13
OA	90.12±0.43	92.35±0.13	93.67±0.15	93.73±0.11	95.01±0.22	97.94±0.20	98.92±0.87	**99.85**±0.04
AA	94.34±0.31	95.99±0.13	96.89±0.20	96.87±0.06	97.51±0.17	98.65±0.25	99.29±0.41	**99.83**±0.05
K(x100)	88.98±0.48	91.47±0.14	92.94±0.17	93.02±0.11	94.44±0.24	97.71±0.22	98.80±0.97	**99.83**±0.04
Time (s.)	2.85±0.48	65.21±0.14	**0.94**±0.17	86.63±0.11	177.78±0.24	177.29±0.22	203.93±0.97	287.58±0.04

close to the original ground-truth. In addition, if we compare the original ResNet to our A-ResNet, we can see that the classification maps produced by the latter exhibit borders between classes that are more sharply defined and clean than those obtained by the original ResNet (for instance, in the SV scene, the A-ResNet provides a better separation between the Fallow-rough-plow field and the Vinyard-vertical-trellis and Grapes-untrained classes).

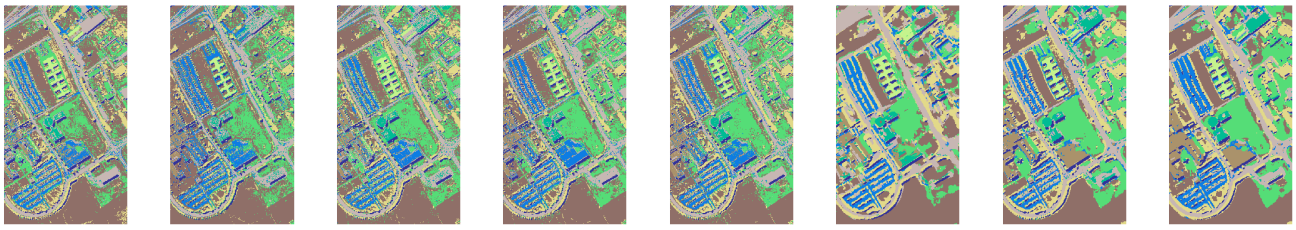
TABLE VII
CLASSIFICATION RESULTS FOR UH DATA SET

Class	RF	MLR	SVM	MLP	CNN1D	CNN2D	ResNet	A-ResNet
Grass healthy	82.49±0.05	**82.62**±0.00	82.34±0.00	81.58±0.38	81.75±0.69	80.48±2.48	82.15±0.47	81.39±1.05
Grass stressed	83.36±0.15	83.93±0.00	83.36±0.00	81.67±0.67	**95.04**±5.33	85.49±2.45	85.09±0.08	84.91±0.49
Grass synthetic	97.82±0.25	99.80±0.00	99.80±0.00	99.64±0.08	**99.88**±0.10	88.99±7.40	98.26±0.52	98.38±0.36
Tree	91.74±0.31	98.01±0.00	**98.96**±0.00	88.69±1.11	89.45±0.59	83.66±3.02	89.55±1.69	86.14±2.20
Soil	96.80±0.20	97.16±0.00	98.77±0.00	97.08±0.43	98.63±0.56	100.00±0.00	100.00±0.00	100.00±0.00
Water	**99.16**±0.28	94.41±0.00	97.90±0.00	94.41±0.00	95.94±1.68	92.59±2.33	95.80±0.00	97.34±1.90
Residential	75.28±0.47	74.25±0.00	77.43±0.00	76.79±2.03	**80.88**±3.59	74.65±3.56	77.28±0.93	76.96±5.42
Commercial	33.01±0.32	65.15±0.00	60.30±0.00	55.82±4.08	80.32±6.54	**80.85**±5.07	79.09±1.14	77.45±3.91
Road	69.40±0.35	69.12±0.00	76.77±0.00	69.91±5.40	77.09±5.76	81.34±3.26	**88.63**±2.35	88.35±1.06
Highway	43.86±0.31	54.44±0.00	61.29±0.00	49.71±3.46	72.57±13.83	63.69±1.40	71.47±10.58	**86.89**±12.40
Railway	70.36±0.25	76.09±0.00	80.55±0.00	75.67±1.37	86.36±6.43	93.74±3.18	**98.14**±1.16	96.28±1.45
Parking lot1	54.77±0.81	73.39±0.00	79.92±0.00	77.16±5.41	91.91±1.68	96.96±2.01	**98.79**±0.31	98.04±1.40
Parking lot2	60.14±0.36	68.42±0.00	70.88±0.00	72.21±2.98	74.74±3.34	**82.88**±3.09	80.42±3.24	79.37±5.21
Tennis court	98.87±0.40	98.79±0.00	100.00±0.00	99.03±0.20	99.36±0.32	98.79±1.33	100.00±0.00	100.00±0.00
Running track	97.50±0.21	95.98±0.00	96.41±0.00	98.31±0.33	98.14±0.49	97.34±3.23	**99.96**±0.08	99.87±0.25
OA	73.09±0.11	79.53±0.00	81.86±0.00	77.98±0.79	86.66±0.44	85.18±0.42	88.20±0.86	**88.71**±0.67
AA	72.16±0.08	76.97±0.00	79.04±0.00	81.18±0.68	88.14±0.35	86.76±0.21	89.64±0.75	**90.09**±0.37
K(x100)	71.09±0.11	77.89±0.00	80.43±0.00	76.29±0.85	85.53±0.47	83.90±0.45	87.18±0.93	**87.73**±0.73
Time (s.)	2.68±0.11	21.25±0.00	**0.37**±0.00	46.09±0.85	94.41±0.47	165.33±0.45	10.44±0.93	34.23±0.73



(a) (b) (c) (d) (e) (f) (g) (h)

Fig. 5. Classification maps provided for the IP data set by different methods (see Table IV). (a) RF (75.31%). (b) MLR (77.76%). (c) SVM (84.48%). (d) MLP (83.50%). (e) CNN1D (84.02%). (f) CNN2D (92.69%). (g) ResNet (95.94%). (h) A-ResNet (98.75%).



(a) (b) (c) (d) (e) (f) (g) (h)

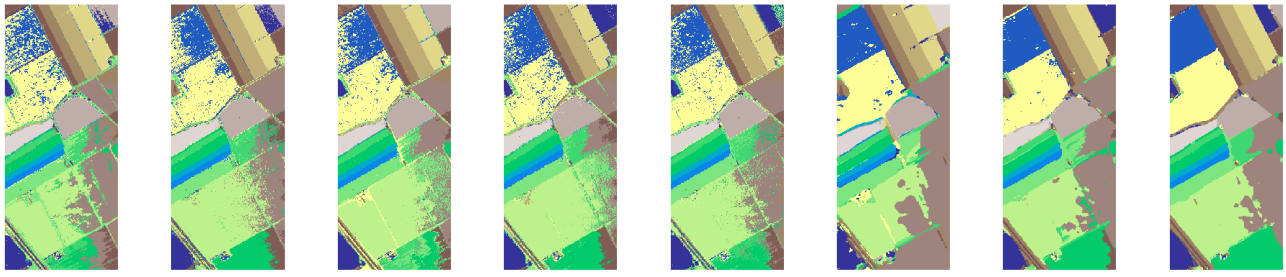
Fig. 6. Classification maps provided for the UP data set by different methods (see Table V). (a) RF (89.37%). (b) MLR (89.73%). (c) SVM (94.10%). (d) MLP (94.04%). (e) CNN1D (94.61%). (f) CNN2D (98.27%). (g) ResNet (99.39%). (h) A-ResNet (99.86%).

2) Experiment 2 (Comparison Between the Advanced Spectral–Spatial HSI Classifiers and the Proposed Method): In order to focus, in more detail, on spectral–spatial classifiers, this experiment compares the proposed attentional model with several spectral–spatial methods discussed in [92]. In this context, the proposed A-Resnet has been adapted to receive the same input data as PCA-CNN, EMP-CNN, and Gabor-CNN, extracting from a fixed training set available for the UP scene [92] the same patches with size $27 \times 27 \times 3$.

The obtained results can be observed in Table VIII. Focusing on the methods described in [92], it is interesting to note that the convolution-based ones are able to reach the highest OA scores, being Gabor-CNN the best one in [92] (thanks to the ability of the Gabor filter to extract and encode highly discriminant spatial features). However, the A-ResNet is able to outperform the OA values of the methods reported in [92], exhibiting 92.06% OA, which is around 0.44% points higher than the Gabor-CNN.

3) Experiment 3 (Evolution of Overall Accuracy of ResNet and A-Resnet When Different Training Ratios Are Considered): Focusing on residual models, the original ResNet and the proposed A-ResNet, this experiment studies the behavior of both models when different amounts of labeled data are available to perform the training step. The IP, UP, and SV scenes have been considered, training the models with 5%, 10%, and 15% of the available labeled samples for the IP scene, and 1%, 5%, and 10% of the available labeled samples for the UP and SV scenes, respectively.

The obtained results are graphically displayed in Fig. 8. We can observe that, when few training samples are used (5% for IP and 1% for UP and SV, respectively), the proposed A-ResNet model is able to reach the best OA values with the

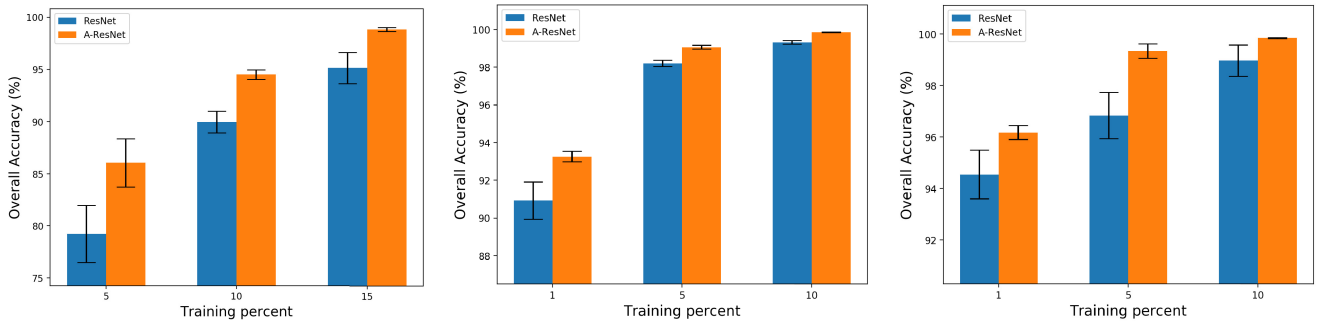


(a) (b) (c) (d) (e) (f) (g) (h)

Fig. 7. Classification maps provided for the SV data set by different methods (see Table VI). (a) RF (90.12%). (b) MLR (92.35%). (c) SVM (93.67%). (d) MLP (93.73%). (e) CNN1D (95.01%). (f) CNN2D (97.94%). (g) ResNet (98.92%). (h) A-ResNet (99.85%).

TABLE VIII
CLASSIFICATION RESULTS FOR UP DATA SET WITH THE FIXED TRAINING SET USED IN [92]

Class	MRF-Gauss	CSVM	EP-CNN	PCA-CNN	EMP-CNN	Gabor-CNN	ResNet	A-ResNet
Asphalt	84.84	92.56	88.43	92.23	**95.87**	87.75	86.53	90.74
Meadows	72.56	73.60	91.64	97.72	**99.50**	97.25	96.96	99.10
Gravel	65.12	71.68	75.95	52.85	61.12	70.92	89.31	**92.51**
Trees	96.63	**98.97**	96.53	89.46	94.81	97.09	93.03	92.89
Painted	99.91	**100.00**	98.56	99.46	95.15	98.83	98.38	97.21
Bare	92.34	**96.35**	57.87	57.66	64.84	64.62	55.36	66.16
Bitumen	91.95	**92.46**	80.43	91.42	80.63	76.66	85.12	82.06
Self-Blocking	94.59	97.41	98.10	98.06	97.26	**99.05**	97.32	96.88
Shadows	**98.99**	95.09	96.84	98.48	96.08	98.36	82.52	81.51
OA	81.78	84.58	87.01	88.93	91.37	91.62	89.45	**92.06**
AA	88.55	**90.90**	87.15	86.37	87.25	87.83	87.03	88.68
K(x100)	76.76	80.31	83.08	85.44	88.67	**89.14**	85.52	89.11



(a) (b) (c)

Fig. 8. Evolution of the OA (Y-axis) for the ResNet and the proposed model (A-ResNet) when classifying (a) IP, (b) UP, and (c) SV hyperspectral scenes using different training ratios.

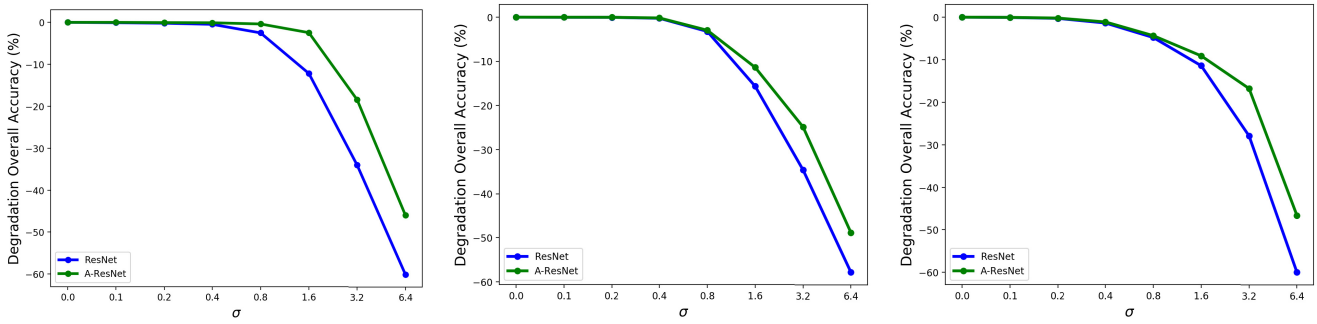
lowest standard deviation, suggesting that the proposed method is able to better address the problem of overfitting when few training samples are provided to the network, obtaining robust results. As we feed more samples to the network, the accuracy gap between the original ResNet and the proposed A-ResNet becomes smaller although the deviation of the attentional network is always much smaller than that of the standard ResNet. This indicates that the proposed method is able to improve the standard ResNet when few training samples are employed, achieving, at least, the same result when a reasonable amount of training samples are used [see Fig. 8(c), obtained using 10% of the available labeled samples for the SV scene].

4) Experiment 4 (Comparison Between the Basic ResNet and the Proposed Method): Motivated by the previous experiment, the fourth experiment studies, in more detail, the behavior of the basic ResNet and the proposed model A-ResNet. The goal of this experiment is to validate the performance and robustness of the proposed method with respect to ResNet when the test data are corrupted. In remote sensing, it is desirable to generate models that process data in a robust manner, for instance, training and testing the classifier model with data obtained at different temporal acquisitions, or after different captures of the same area. These situations introduce certain disturbances or changes in the training and testing data to which the models must be able to respond in a reliable

TABLE IX

OA OF RESNET AND THE PROPOSED MODEL (A-RESNET) OVER THE IP, UP, AND SV DATA SETS WHEN DIFFERENT NORMAL RANDOM PERTURBATIONS ARE INSERTED INTO THE DATA

Normal Perturbation	$\mu = 0, \sigma = 0.10$		$\mu = 0, \sigma = 0.20$		$\mu = 0, \sigma = 0.40$		$\mu = 0, \sigma = 0.80$		$\mu = 0, \sigma = 1.60$		$\mu = 0, \sigma = 3.20$		$\mu = 0, \sigma = 6.40$	
Dataset	**ResNet**	**A-ResNet**	**ResNet**	**A-ResNet**	**ResNet**	**A-ResNet**	**ResNet**	**A-ResNet**	**ResNet**	**A-ResNet**	**ResNet**	**A-ResNet**	**ResNet**	**A-ResNet**
IP	95.05	**98.84**	94.93	**98.78**	94.67	**98.74**	92.63	**98.45**	82.97	**96.38**	61.13	**80.40**	35.06	**52.87**
UP	99.29	**99.85**	99.28	**99.84**	99.07	**99.69**	96.09	**96.9**	83.67	**88.49**	64.74	**75.01**	41.57	**51.08**
SV	98.90	**99.81**	98.69	**99.64**	97.60	**98.73**	94.19	**95.51**	87.57	**90.77**	71.08	**83.1**	39.05	**53.28**



(a) (b) (c)

Fig. 9. Degradation of the OA (Y-axis) of ResNet and the proposed model A-ResNet for (a) IP, (b) UP, (c) SV, and (d) KSC, comparing the accuracy reached with the original data ($\sigma = 0$) and the accuracy reached with perturbed data, being $\sigma = \{0.01, 0.02, 0.03, 0.04, 0.05\}$ (X-axis).

manner. As a result, this experiment evaluates how Resnet and A-ResNet behave when they have to deal with perturbed data.

In order to simulate perturbed data, the original IP, UP, and SV data sets have been modified through a random normal distribution with mean $\mu = 0$ and seven different standard deviation values $\sigma = \{0.10, 0.20, 0.40, 0.80, 1.60, 3.20, 6.40\}$. Neural models have been trained over the original data sets using 15% of the available labeled samples from IP and 10% of the available labeled samples from UP and SV. Again, patches of $11 \times 11 \times 40$ have been employed as the input data. The obtained results are given in Table IX.

With slight disturbances ($\sigma = 0.10$), we can observe that the ResNet exhibits a small decay of OA values in comparison with the case that no perturbations are present in the IP (-0.89) and UP (-0.1) data sets, while in the SV data set, the difference is very small (-0.02), as we can observe in Fig. 9. In turn, the A-ResNet is not significantly affected by the introduced perturbations. For instance, in the IP scene, it is even able to outperform the ResNet in terms of OA, being 0.09% points better when noise is not included.

However, as the noise level increases, we can see how the OA of the standard ResNet decreases significantly, in particular, from $\sigma = 1.6$. Therefore, the features extracted by the standard ResNet from these data sets are not relevant or generic enough to be applied in scenarios with perturbations. Instead of that, the performance of the proposed models remains more stable. For instance, for the IP data set, the A-ResNet exhibits a degradation of 2.37% points, while the ResNet exhibits a degradation of 12.97 points. Also, in the experiments with the UP and SV scenes, the ResNet is more affected than the A-ResNet although the gap between the two seems smaller. However, with greater σ values, the gap becomes larger. This behavior can be also observed for the rest of σ values (see Fig. 9); ResNet reaches the lowest OA and exhibits the worse degradation of performance with perturbed data, while the proposed model maintains a high OA and significantly lower degradation.

The OA values in Table IX and the degradation performance in Fig. 9 indicate that the proposed model is more robust to perturbations in the data, achieving high OA values. Also, it is able to extract more discriminative features from the original training data in comparison with ResNet, being the A-ResNet the most robust architecture for all data sets (even in the presence of significant distortions).

V. CONCLUSION

In this paper, a new model for spatial–spectral HSI classification has been proposed by combining a DL architecture (ResNet) and visual attention techniques. The filtering system introduced by the visual attention model, following bottom-up and top-down visual selections, allows for postprocessing of the extracted data, enhancing the quality of the feature extraction process as well as obtaining more representative and significant features, leading to a more precise and robust classification of HSI data.

Our experimental comparisons have been conducted using four publicly available HSI data sets, evaluating the proposed visual attention-driven model (A-ResNet) versus seven standard machine learning and DL classifiers and six advanced spectral–spatial methods, revealing that the proposed networks exhibit competitive results when compared to state-of-the-art techniques, such as CNNs (combined with different techniques) and ResNets. Also, a deeper comparison between the ResNet and the proposed model with different amounts of training data and perturbed data revealed that our newly proposed model is able to extract more relevant, discriminative, and complete features from HSI scenes, exhibiting robustness to network degradation when very limited training samples and/or highly disturbed data are considered.

As future work, we intend to improve the parameter optimization mechanism of the proposed network (particularly when very few labeled samples are available) in order to

reduce the effect of overfitting. Also, we are planning to combine additional visual attention techniques with other deep models, with the aim of enhancing the quality of the extracted features and the final classification results.

ACKNOWLEDGMENT

The authors would like to thank the Associate Editor and the three anonymous reviewers for their outstanding comments and suggestions, which greatly helped us to improve the technical quality and presentation of this paper.

REFERENCES

[1] P. Ghamisi *et al.*, "Advances in hyperspectral image and signal processing: A comprehensive overview of the state of the art," *IEEE Geosci. Remote Sens. Mag.*, vol. 5, no. 4, pp. 37–78, Dec. 2017.

[2] D. Chutia, D. K. Bhattacharyya, K. K. Sarma, R. Kalita, and S. Sudhakar, "Hyperspectral remote sensing classifications: A perspective survey," *Trans. GIS*, vol. 20, no. 4, pp. 463–490, 2016.

[3] 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.

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

[5] D. Landgrebe, "Hyperspectral image data analysis," *IEEE Signal Process. Mag.*, vol. 19, no. 1, pp. 17–28, Jan. 2002.

[6] D. Haboudane, J. R. Miller, E. Pattey, P. J. Zarco-Tejada, and I. B. Strachan, "Hyperspectral vegetation indices and novel algorithms for predicting Green LAI of crop canopies: Modeling and validation in the context of precision agriculture," *Remote Sens. Environ.*, vol. 90, no. 3, pp. 337–352, 2004.

[7] 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 J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 9, no. 9, pp. 4117–4128, Sep. 2016.

[8] S. L. Martin and T. George, "Applications of hyperspectral image analysis for precision agriculture," *Proc. SPIE*, vol. 10639, May 2018, Art. no. 1063916.

[9] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. M. 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.

[10] P. W. Yuen and M. Richardson, "An introduction to hyperspectral imaging and its application for security, surveillance and target acquisition," *Imag. Sci. J.*, vol. 58, no. 5, pp. 241–253, Oct. 2010.

[11] E. Puckrin, C. S. Turcotte, M.-A. Gagnon, J. Bastedo, V. Farley, M. Chamberland, "Airborne infrared hyperspectral imager for intelligence, surveillance, and reconnaissance applications," *Proc. SPIE*, vol. 8360, 2012, Art. no. 836004. doi: 10.1117/12.918251.

[12] B. Uzkent, A. Rangnekar, and M. Hoffman, "Aerial vehicle tracking by adaptive fusion of hyperspectral likelihood maps," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 233–242.

[13] G. A. Carter *et al.*, "Remote sensing and mapping of tamarisk along the Colorado River, USA: A comparative use of summer-acquired hyperion, thematic mapper and quickbird data," *Remote Sens.*, vol. 1, no. 3, pp. 318–329, 2009.

[14] P. Ghamisi, J. Plaza, Y. Chen, J. Li, and A. J. Plaza, "Advanced spectral classifiers for hyperspectral images: A review," *IEEE Geosci. Remote Sens. Mag.*, vol. 5, no. 1, pp. 8–32, Mar. 2017.

[15] J. M. Haut, M. Paoletti, J. Plaza, and A. Plaza, "Cloud implementation of the K-means algorithm for hyperspectral image analysis," *J. Supercomput.*, vol. 73, no. 1, pp. 514–529, 2017.

[16] J.-M. Yang, P.-T. Yu, and B.-C. Kuo, "A nonparametric feature extraction and its application to nearest neighbor classification for hyperspectral image data," *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 3, pp. 1279–1293, Mar. 2010.

[17] J. A. Gualtieri and R. F. Cromp, "Support vector machines for hyperspectral remote sensing classification," in *Proc. 27th AIPR Workshop, Adv. Comput.-Assist. Recognit.*, vol. 3584, 1999, pp. 221–233. doi: 10.1117/12.339824.

[18] 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.

[19] G. Mercier and M. Lennon, "Support vector machines for hyperspectral image classification with spectral-based kernels," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, vol. 1, Jul. 2003, pp. 288–290.

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

[21] J. Haut, M. Paoletti, A. Paz-Gallardo, J. Plaza, and A. Plaza, "Cloud implementation of logistic regression for hyperspectral image classification," in *Proc. 17th Int. Conf. Comput. Math. Methods Sci. Eng. (CMMSE)*, J. Vigo-Aguiar, Ed. Cádiz, Spain, 2017, pp. 1063–2321.

[22] 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.

[23] R. E. Bellman, *Adaptive Control Processes: A Guided Tour*, vol. 2045. Princeton, NJ, USA: Princeton Univ. Press, 2015.

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

[25] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.

[26] W. Li *et al.*, "Stacked Autoencoder-based deep learning for remote-sensing image classification: A case study of African land-cover mapping," *Int. J. Remote Sens.*, vol. 37, no. 23, pp. 5632–5646, 2016.

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

[28] Z. Lin, Y. Chen, X. Zhao, and G. Wang, "Spectral-spatial classification of hyperspectral image using autoencoders," in *Proc. 9th Int. Conf. Inf., Commun. Signal Process.*, Dec. 2013, pp. 1–5.

[29] 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.

[30] Y. Chen, X. Zhao, and X. Jia, "Spectral-spatial classification of hyperspectral data based on deep belief network," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 8, no. 6, pp. 2381–2392, Jun. 2015.

[31] C. Tao, H. Pan, Y. Li, and Z. Zou, "Unsupervised spectral–spatial feature learning with stacked sparse autoencoder for hyperspectral imagery classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 12, pp. 2438–2442, Dec. 2015.

[32] Y. Liu, G. Cao, Q. Shen, and M. Siegel, "Hyperspectral classification via deep networks and superpixel segmentation," *Int. J. Remote Sens.*, vol. 36, no. 13, pp. 3459–3482, Jul. 2015.

[33] L. Wang, J. Zhang, P. Liu, K.-K. R. Choo, and F. Huang, "Spectral–spatial multi-feature-based deep learning for hyperspectral remote sensing image classification," *Soft Comput.*, vol. 21, no. 1, pp. 213–221, Jan. 2017.

[34] H. Luo, Y. Y. Tang, X. Yang, L. Yang, and H. Li, "Autoencoder with extended morphological profile for hyperspectral image classification," in *Proc. 3rd IEEE Int. Conf. (CYBCONF)*, Jun. 2017, pp. 1–4.

[35] S. Paul and D. N. Kumar, "Spectral-spatial classification of hyperspectral data with mutual information based segmented stacked autoencoder approach," *ISPRS J. Photogram. Remote Sens.*, vol. 138, pp. 265–280, Apr. 2018.

[36] Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio, "Object recognition with gradient-based learning," in *Shape, Contour and Grouping in Computer Vision*. London, U.K.: Springer-Verlag, 1999, p. 319. [Online]. Available: http://dl.acm.org/citation.cfm?id=646469.691875

[37] M. E. Paoletti, J. M. Haut, J. Plaza, and A. Plaza, "A new deep convolutional neural network for fast hyperspectral image classification," *ISPRS J. Photogram. Remote Sens.*, vol. 145, pp. 120–147, Nov. 2018. doi: 10.1016/j.isprsjprs.2017.11.021.2017.

[38] 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.

[39] 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.

[40] 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, Feb. 2018.

[41] 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, Feb. 2019.

[42] W. Wang, S. Dou, Z. Jiang, and L. Sun, "A fast dense spectral–spatial convolution network framework for hyperspectral images classification," *Remote Sens.*, vol. 10, no. 7, p. 1068, 2018.

[43] M. E. Paoletti, J. M. Haut, J. Plaza, and A. Plaza, "Deep&dense convolutional neural network for hyperspectral image classification," *Remote Sens.*, vol. 10, no. 9, p. 1454, 2018.

[44] F. Deng, S. Pu, X. Chen, Y. Shi, T. Yuan, and S. Pu, "Hyperspectral image classification with capsule network using limited training samples," *Sensors*, vol. 18, no. 9, p. 3153, 2018.

[45] M. E. Paoletti *et al.*, "Capsule networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 4, pp. 2145–2160, Apr. 2019.

[46] T. S. Nazaré, G. B. P. da Costa, W. A. Contato, and M. Ponti, "Deep convolutional neural networks and noisy images," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, M. Mendoza and S. Velastín, Eds. Cham, Switzerland: Springer, 2018, pp. 416–424.

[47] N. Imamoglu *et al.*, "Hyperspectral image dataset for benchmarking on salient object detection," in *Proc. 10th Int. Conf. Qual. Multimedia Exper. (QoMEX)*, May 2018, pp. 1–3.

[48] A. Borji, H. R. Tavakoli, and Z. Bylinskii, "Bottom-up attention, models of," 2018, *arXiv:1810.05680*. [Online]. Available: https://arxiv.org/abs/1810.05680

[49] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 11, pp. 1254–1259, Nov. 1998.

[50] X. Chen, H. Huoa, F. Taoa, D. Lib, and Z. Lia, "A computational method to emulate bottom-up attention to remote sensing images," in *Proc. 21st Congr. Int. Soc. Photogram. Remote Sens. (ISPRS)*, vol. 37, 2008, pp. 244–249.

[51] L. Zhang, H. Li, P. Wang, and X. Yu, "Detection of regions of interest in a high-spatial-resolution remote sensing image based on an adaptive spatial subsampling visual attention model," *GISci. Remote Sens.*, vol. 50, no. 1, pp. 112–132, 2013.

[52] L. Zhang and K. Yang, "Region-of-interest extraction based on frequency domain analysis and salient region detection for remote sensing image," *IEEE Geosci. Remote Sens. Lett.*, vol. 11, no. 5, pp. 916–920, May 2014.

[53] L. Zhang, K. Yang, and H. Li, "Regions of interest detection in panchromatic remote sensing images based on multiscale feature fusion," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 12, pp. 4704–4716, Dec. 2014.

[54] D. Zhu, B. Wang, and L. Zhang, "Airport target detection in remote sensing images: A new method based on two-way saliency," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 5, pp. 1096–1100, May 2015.

[55] L. Zhang and A. Li, "Region-of-interest extraction based on saliency analysis of co-occurrence histogram in high spatial resolution remote sensing images," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 8, no. 5, pp. 2111–2124, May 2015.

[56] T. Li, J. Zhang, X. Lu, and Y. Zhang, "SDBD: A hierarchical region-of-interest detection approach in large-scale remote sensing image," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 5, pp. 699–703, May 2017.

[57] Z. Li and L. Itti, "Saliency and gist features for target detection in satellite images," *IEEE Trans. Image Process.*, vol. 20, no. 7, pp. 2017–2029, Jul. 2011.

[58] F. Bi, B. Zhu, L. Gao, and M. Bian, "A visual search inspired computational model for ship detection in optical satellite images," *IEEE Geosci. Remote Sens. Lett.*, vol. 9, no. 4, pp. 749–753, Jul. 2012.

[59] X. Ke and G. He, "Visual attention based model for target detection in high resolution remote sensing images," in *Proc. Int. Conf. Comput. Vis. Remote Sens. (CVRS)*, Dec. 2012, pp. 84–89.

[60] C. Wang, X. Bai, S. Wang, J. Zhou, and P. Ren, "Multiscale visual attention networks for object detection in VHR remote sensing images," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 2, pp. 310–314, Feb. 2019.

[61] S. Kumar, J. Ghosh, and M. M. Crawford, "Best-bases feature extraction algorithms for classification of hyperspectral data," *IEEE Trans. Geosci. Remote Sens.*, vol. 39, no. 7, pp. 1368–1379, Jul. 2001.

[62] Q. Wang, J. Lin, and Y. Yuan, "Salient band selection for hyperspectral image classification via manifold ranking," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 6, pp. 1279–1289, Jun. 2016.

[63] J. Ba, V. Mnih, and K. Kavukcuoglu, "Multiple object recognition with visual attention," 2014, *arXiv:1412.7755*. [Online]. Available: https://arxiv.org/abs/1412.7755

[64] T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, and Z. Zhang, "The application of two-level attention models in deep convolutional neural network for fine-grained image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 842–850.

[65] F. Wang *et al.*, "Residual attention network for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6450–6458.

[66] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[67] T. N. Wiesel and D. H. Hubel, "Receptive fields of single neurones in the cat's striate cortex," *J. Physiol.*, vol. 148, no. 3, pp. 574–591, 1959.

[68] L. Zhang, L. Zhang, and B. Du, "Deep learning for remote sensing data: A technical tutorial on the state of the art," *IEEE Geosci. Remote Sens. Mag.*, vol. 4, no. 2, pp. 22–40, Jun. 2016.

[69] Q. Du and J. E. Fowler, "Hyperspectral image compression using JPEG2000 and principal component analysis," *IEEE Geosci. Remote Sens. Lett.*, vol. 4, no. 2, pp. 201–205, Apr. 2007.

[70] J. M. Haut, M. E. Paoletti, J. Plaza, and A. Plaza, "Fast dimensionality reduction and classification of hyperspectral images with extreme learning machines," *J. Real-Time Image Process.*, vol. 15, no. 3, pp. 439–462, 2018.

[71] 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.

[72] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural Comput.*, vol. 29, no. 9, pp. 2352–2449, Sep. 2017.

[73] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.

[74] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," in *Advances in Neural Information Processing Systems*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2015, pp. 2377–2385.

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

[76] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980

[77] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. ICML*, vol. 30, no. 1, Jun. 2013, p. 3.

[78] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[79] 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. (CVPR)*, Jul. 2017, pp. 5987–5995.

[80] S. Zagoruyko and N. Komodakis, "Wide residual networks," 2016, *arXiv:1605.07146*. [Online]. Available: https://arxiv.org/abs/1605.07146

[81] D. Han, J. Kim, and J. Kim, "Deep pyramidal residual networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6307–6315.

[82] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Computer Vision—ECCV*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham, Switzerland: Springer, 2016, pp. 630–645. doi: 10.1007/978-3-319-46493-0_38.

[83] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," 2016, *arXiv:1602.07261*. [Online]. Available: https://arxiv.org/abs/1602.07261

[84] J. M. Haut, R. Fernandez-Beltran, M. E. Paoletti, J. Plaza, A. Plaza, and F. Pla, "A new deep generative network for unsupervised remote sensing single-image super-resolution," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 11, pp. 6792–6810, Nov. 2018.

[85] V. A. F. Lamme, H. Supèr, and H. Spekreijse, "Feedforward, horizontal, and feedback processing in the visual cortex," *Current Opinion Neurobiol.*, vol. 8, no. 4, pp. 529–535, 1998.

[86] A. Mahdi and J. Qin, "DeepFeat: A bottom up and top down saliency model based on deep features of convolutional neural nets," 2017, *arXiv:1709.02495*. [Online]. Available: https://arxiv.org/abs/1709.02495

[87] M. Lin, Q. Chen, and S. Yan, "Network in network," 2013, *arXiv:1312.4400*. [Online]. Available: https://arxiv.org/abs/1312.4400

[88] R. O. Green *et al.*, "Imaging spectroscopy and the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS)," *Remote Sens. Environ.*, vol. 65, no. 3, pp. 227–248, Sep. 1998. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0034425798000649

[89] 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," *Proc. SPIE*, vol. 868, pp. 134–142, Apr. 1988.

[90] X. Xu, F. Lil, 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.
[91] C. Debes *et al.*, "Hyperspectral and LiDAR data fusion: Outcome of the 2013 GRSS data fusion contest," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 6, pp. 2405–2418, Jun. 2014.
[92] P. Ghamisi *et al.*, "New frontiers in spectral-spatial hyperspectral image classification: The latest advances based on mathematical morphology, Markov random fields, segmentation, sparse representation, and deep learning," *IEEE Geosci. Remote Sens. Mag.*, vol. 6, no. 3, pp. 10–43, Sep. 2018.
[93] P. Gurram and H. Kwon, "Contextual SVM using Hilbert space embedding for hyperspectral classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 10, no. 5, pp. 1031–1035, Sep. 2013.
[94] P. Ghamisi, B. Höfle, and X. X. Zhu, "Hyperspectral and LiDAR data fusion using extinction profiles and deep convolutional neural network," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 10, no. 6, pp. 3011–3024, Jun. 2017.

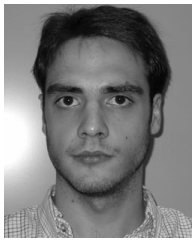
Juan Mario Haut (S'17) received the B.Sc. and M.Sc. degrees in computer engineering from the University of Extremadura, Cáceres, Spain, in 2011 and 2014, respectively, where he is currently pursuing the Ph.D. degree with the University Teacher Training Programme from the Spanish Ministry of Education.

He is currently a member of the Hyperspectral Computing Laboratory, Department of Computers and Communications, University of Extremadura. His research interests include remote sensing and analysis of very high spectral resolution with the current focus on deep learning and cloud computing.

Mr. Haut was a recipient of the recognition of Best Reviewers of the IEEE GEOSCIENCE AND REMOTE SENSING LETTERS in 2019.

Mercedes E. Paoletti (S'17) received the B.Sc. and M.Sc. degrees in computer engineering from the University of Extremadura, Cáceres, Spain, in 2014 and 2016, respectively, where she is currently pursuing the Ph.D. degree with the University Teacher Training Programme from the Spanish Ministry of Education.

She is currently a member of the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, University of Extremadura. Her research interests include remote sensing and analysis of very high spectral resolution with the current focus on deep learning and high-performance computing.



Javier Plaza (M'09–SM'15) received the M.Sc. and Ph.D. degrees in computer engineering from the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, University of Extremadura, Cáceres, Spain, in 2004 and 2008, respectively.

He is currently a member of the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, University of Extremadura. He has authored more than 150 publications, including over 50 JCR journal papers, ten book chapters, and 90 peer-reviewed conference proceeding papers. His main research interests include hyperspectral data processing and parallel computing of remote sensing data.

Dr. Plaza was a recipient of the Outstanding Ph.D. Dissertation Award at the University of Extremadura in 2008. He was also a recipient of the Best Column Award of the *IEEE Signal Processing Magazine* in 2015 and the Most Highly Cited Paper (2005–2010) in the *Journal of Parallel and Distributed Computing*. He received best paper awards at the IEEE International Conference on Space Technology and the IEEE Symposium on Signal Processing and Information Technology. He has guest edited four special issues on hyperspectral remote sensing for different journals. He is also an Associate Editor of the IEEE GEOSCIENCE AND REMOTE SENSING LETTERS and an Associate Editor of the IEEE Remote Sensing Code Library. Additional information: http://www.umbc.edu/rssipl/people/jplaza.

Antonio Plaza (M'05–SM'07–F'15) received the M.Sc. and Ph.D. degrees in computer engineering from the Head of the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, University of Extremadura, Cáceres, Spain, in 1999 and 2002, respectively.

He is currently the Head of the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, University of Extremadura. He has authored more than 600 publications, including over 200 JCR journal papers (over 160 in IEEE journals), 23 book chapters, and around 300 peer-reviewed conference proceeding papers. His main research interests include hyperspectral data processing and parallel computing of remote sensing data.

Dr. Plaza was a member of the Steering Committee of the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING (JSTARS). He is also a fellow of the IEEE for his contributions to hyperspectral data processing and parallel computing of earth observation data. He was a recipient of the Recognition of Best Reviewer of the IEEE GEOSCIENCE AND REMOTE SENSING LETTERS in 2009 and the Recognition of Best Reviewer of the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING in 2010, for which he has served as an Associate Editor from 2007 to 2012. He was also a recipient of the Most Highly Cited Paper (2005–2010) in the *Journal of Parallel and Distributed Computing*, the 2013 Best Paper Award of the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING (JSTARS), and the Best Column Award of the *IEEE Signal Processing Magazine* in 2015. He received the best paper awards at the IEEE International Conference on Space Technology and the IEEE Symposium on Signal Processing and Information Technology. He has guest edited ten special issues on hyperspectral remote sensing for different journals. He is also an Associate Editor of the IEEE ACCESS (receiving a recognition as an Outstanding Associate Editor of the journal in 2017), and was a member of the Editorial Board of the IEEE Geoscience and Remote Sensing Newsletter from 2011 to 2012 and the *IEEE Geoscience and Remote Sensing Magazine* in 2013. He served as the Director of Education Activities for the IEEE Geoscience and Remote Sensing Society (GRSS) from 2011 to 2012 and the President of the Spanish Chapter of IEEE GRSS from 2012 to 2016. He reviewed more than 500 manuscripts of over 50 different journals. He has served as the Editor-in-Chief of the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING from 2013 to 2017. Additional information: http://www.umbc.edu/rssipl/people/aplaza.

Jun Li (SM'16) was born in Lodi, Hunan, China, in 1982. She received the Geographical Information Systems degree from Hunan Normal University, Changsha, China, in 2004, the M.Sc. degree in remote sensing and photogrammetry from Peking University, Beijing, China, in 2007, and the Ph.D. degree in electrical and computer engineering from Instituto Superior Técnico, Technical University of Lisbon, Lisbon, Portugal, in 2011.

From 2011 to 2012, she was a Post-Doctoral Researcher with the Department of Technology of Computers and Communications, University of Extremadura, Cáceres, Spain. She is currently a Professor with the School of Geography and Planning, Sun Yat-sen University, Guangzhou, China, where she founded her own research group on hyperspectral image analysis in 2013. She has published a total of 69 journal citation report (JCR) papers, 48 conference international conference papers, and one book chapter. Her main research interests include remotely sensed hyperspectral image analysis, signal processing, supervised/semisupervised learning, and active learning.

Dr. Li received a significant number of citations to her published works, with several papers distinguished as "Highly Cited Papers" in Thomson Reuters' Web of Science-Essential Science Indicators (WoS-ESI). She has obtained several prestigious funding grants at the national and international level. Her students have also obtained important distinctions and awards at international conferences and symposia. She has served as the Guest Editor for the Special Issue of the prestigious PROCEEDINGS OF THE IEEE journal. She has also served as the Guest Editor for a Special Issue of the prestigious *ISPRS Journal of Photogrammetry and Remote Sensing* journal. She has been serving as an Associate Editor for the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING (JSTARS) since 2014.

Escuela Politecnica
Av. de la Universidad, S/N, 10003
Caceres, Spain
Phone: 0034927257000. Ext. 51662
Email: aplaza,jplaza{@unex.es}

Dr. Antonio Plaza Miguel y Dr. Javier Plaza Miguel como directores de la tesis titulada "Procesamiento eficiente y profundo de imgenes hiperespectrales de la observación remota de la Tierra y aplicaciones en tareas de clasificación", certifico el factor de impacto y la categorización de la siguiente publicación, incluida en la tesis doctoral. Del mismo modo, se especifica la aportación del doctorado. Si necesitas cualquier información o clarificación, por favor, no dude en contactar conmigo.

Antonio Plaza Miguel PhD. and Javier Plaza Miguel PhD as directors of the Phd thesis titled "Deep-efficient processing of remote sensing hyperspectral images and applications in classification tasks.", certify the impact factor and the categorization of the following publication, included in the doctoral thesis. In the same way, the contribution of the doctorate is specified. If you need any further information or clarification, please do not hesitate contacting me.

Artculo / Paper

Autores/Authors: **M. E. Paoletti**, J. M. Haut, J. Plaza and A. Plaza.
Title: Neural Ordinary Differential Equations for Hyperspectral Image Classification.
Journal: IEEE Transactions on Geoscience and Remote Sensing.
Other Information: vol. 58, no. 3, pp. 1718-1734, March 2020.
DOI: 10.1109/TGRS.2019.2948031.
Impact factor 2018: 5.630. Q1
Abstract: Advances in deep learning (DL) have allowed for the development of more complex and powerful neural architectures. The adoption of deep convolutional-based architectures with residual learning [residual networks (ResNets)] has reached the state-of-the-art performance in hyperspectral image (HSI) classification. Traditionally, ResNets have been considered as stacks of discrete layers, where each one obtains a hidden state of the input data. This formulation must deal with very deep networks, which suffer from an important data degradation as they become deeper. Moreover, these complex models exhibit significant requirements in terms of memory due to the amount of parameters that need to be fine tuned. This leads to inadequate generalization and loss of accuracy. In order to address these issues, this article redesigns the ResNet as a continuous-time evolving model, where hidden representations (or states) are obtained with respect to time (understood as the depth of the network) through the evaluation of an ordinary differential equation (ODE), which is combined with a deep neural architecture. Our experimental results, conducted with four well-known HSI data sets, indicate that redefining deep networks as continuous systems through ODEs offers flexibility when processing and classifying these kinds of remotely sensed data, achieving significant performance even when a very few training samples are available.

Contribución del doctorado: Planteamiento de la hipótesis, desarrollo práctico, análisis y discusión de los resultados, elaboración y escritura del manuscrito.

Firma / Signature
Mayo / May, 2020

Antonio Plaza Miguel

Javier Plaza Miguel

Neural Ordinary Differential Equations for Hyperspectral Image Classification

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

***Abstract*— Advances in deep learning (DL) have allowed for the development of more complex and powerful neural architectures. The adoption of deep convolutional-based architectures with residual learning [residual networks (ResNets)] has reached the state-of-the-art performance in hyperspectral image (HSI) classification. Traditionally, ResNets have been considered as stacks of discrete layers, where each one obtains a hidden state of the input data. This formulation must deal with very deep networks, which suffer from an important data degradation as they become deeper. Moreover, these complex models exhibit significant requirements in terms of memory due to the amount of parameters that need to be fine tuned. This leads to inadequate generalization and loss of accuracy. In order to address these issues, this article redesigns the ResNet as a continuous-time evolving model, where hidden representations (or states) are obtained with respect to time (understood as the depth of the network) through the evaluation of an ordinary differential equation (ODE), which is combined with a deep neural architecture. Our experimental results, conducted with four well-known HSI data sets, indicate that redefining deep networks as continuous systems through ODEs offers flexibility when processing and classifying these kinds of remotely sensed data, achieving significant performance even when a very few training samples are available.**

***Index Terms*— Deep learning (DL), hyperspectral images (HSIs), ordinary differential equations (ODEs), residual networks (ResNets).**

Manuscript received March 14, 2019; revised August 10, 2019; accepted October 13, 2019. Date of publication November 6, 2019; date of current version February 26, 2020. This work was supported in part by the 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, and 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, and de divulgación y de transferencia de conocimiento por los Grupos de Investigación de Extremadura) under Grant GR18060, in part by the European Union's Horizon 2020 Research and Innovation Programme under Grant 734541 (EOXPOSURE), and in part by Ministerio de Economía y Empresa (MINECO) Project under Grant TIN2015-63646-C5-5-R. *(Corresponding author: Mercedes E. Paoletti.)*

The authors are with the 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; 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.2019.2948031

I. Introduction

REMOTE sensing techniques have been widely employed for detecting, measuring, and monitoring the physical behavior and/or characteristics of large areas of the earth through the acquisition and measurement of radiation emitted or reflected by the terrestrial materials that comprise the observed surfaces, which are captured by specific sensors located on airborne or spaceborne platforms [1]. The interpretation of the obtained measurements can be beneficial to human activity [2], [3]. There is a wide range of remote sensing data, where each one exhibits different spatial and spectral properties depending on the type of employed sensor and measured radiation. Moreover, current earth observation missions are already collecting an extremely large volume of remotely sensed data from satellites and airborne systems [4]. Hyperspectral images (HSIs) are collected by passive spectrometers that measure the reflected solar radiation from the observed areas, creating huge data cubes comprised of hundreds of narrow and continuous spectral wavelengths. As a result, an HSI given by $\mathbf{X} \in \mathbb{R}^{n_1 \times n_2 \times n_{\text{bands}}}$ is comprised of two spatial components that determine the image's width and height ($n_1 \times n_2$) and one spectral component that indicates the number of channels or spectral bands (n_{bands}). As a result, each pixel of $\mathbf{X}$ can be interpreted as a detailed spectral signature or spectral vector $\mathbf{x}_i \in \mathbb{R}^{n_{\text{bands}}} = \{x_{i,1}, \ldots, x_{i,n_{\text{bands}}}\}$, which allows for an accurate characterization of the surface materials [5]. This has attracted the attention of many researchers who employ HSIs into a wide range of applications, including precision agriculture [6], environment and natural resources' management [7], mineralogy [8], forestry [9], disaster monitoring [10], urban planning [11], and defense applications [12], among others.

A large variety of algorithms have been developed to process and extract useful information from HSI data cubes. In this regard, HSI classification methods can greatly benefit from the rich spectral information contained in each pixel $\mathbf{x}_i$. In fact, the classification of these images aims to assign a single category (or label) to each pixel in the image. In mathematical fashion, the goal of a classifier is to approximate a mapping function of the form $f(\cdot, \theta)$, which depends on parameter θ, to map the pixels in the original HSI $\mathcal{X} \subset \mathbb{R}^{n_{\text{samples}}}$ to those labels contained in a set of categories $\mathcal{Y} \subset \mathbb{N}$, i.e., $f : \mathcal{X} \rightarrow \mathcal{Y}$. In the particular case of HSI classification, the procedure consists of mapping each pixel $\mathbf{x}_i$ in $\mathbf{X} \equiv \{\mathbf{x}_1, \ldots, \mathbf{x}_{n_{\text{samples}}}\}$ (with $n_{\text{samples}} = n_1 \cdot n_2$) to

a unique numerical label of n_{classes} possible classes $y_i = \{1, \ldots, n_{\text{classes}}\}$ extracted from the set $Y \equiv \{y_1, \ldots, y_{n_{\text{samples}}}\}$, creating pairs of $\{\mathbf{x}_i, y_i\}_{i=1}^{n_{\text{samples}}}$ for each spectral pixel.

Traditional HSI classification methods are based on the analysis of the each pixel $\mathbf{x}_i$ independently, without considering spatial information, for instance, unsupervised clustering techniques, such as k-means [13], and supervised or semisupervised methods, such as the widely used support vector machine (SVM) [14] or multinomial logistic regression (MLR) [15], among others [16], [17]. Artificial neural networks (ANNs) [18], [19] have acquired great popularity due to their flexibility concerning learning modes (unsupervised, supervised, and semisupervised) and available architectures (shallow, deep, fully, or local connected). Moreover, ANNs work as universal approximators [20], [21], being able to extract representative features and to discover nonlinear relationships from the input data.

Advances in deep learning (DL) [22], [23] have allowed for the implementation of deeper and more complex ANNs, known as deep neural networks (DNNs). These networks are comprised of groups of neurons organized into a hierarchy of multiple nonlinear layers, which are stacked one by one. As a result, DNNs are comprised of one input and one output layer with several hidden layers in-between them. The original data go through the hierarchy of layers, where a different level of data representation is obtained at each layer. These representations are comprised of highly expressive features that encode complex patterns and nonlinear relationships in the data. At the end of the network, highly abstract and discriminative information is obtained, which can be employed to enhance classification tasks. In the following, we briefly review some recent DL works in the literature (focusing on those based on convolutional and residual architectures for HSI data classification), and then, we discuss some shortcomings and limitations of these works and the solutions adopted in this article.

A. Recent Trends in DL for HSI Classification

DNNs traditionally follow a biological neural model, implementing a fully connected (FC) topology where all the neurons in a layer are totally connected with all the neurons of the previous and following layers, as in the multilayer perceptron (MLP). In this way, each neuron applies a dot product between the outputs of previous neurons and the connection weights, simulating synaptic weights. The obtained result is filtered by a threshold function, also known as nonlinear activation function, which encodes the nonlinearities of the data and triggers (or not) the activation of a given neuron. In fact, the DNN approaches adopt the same strategy as traditional pixelwise classifiers. For HSI data, they take as input the spectral pixels of the HSI data cube [18]. In this regard, spectral-based DNNs are quite sensitive to variations in the spectral signatures. It should be noted that HSI data are characterized by their high intraclass variability and interclass similarity (due to perturbations and disturbances in the data collection process at the spectrometer, atmospheric conditions, and so on). Also, HSI data normally exhibit low spatial resolution, which means that a single pixel often contains multiple materials, resulting in mixed spectral signatures. These shortcomings, coupled with the curse of dimensionality and the Hughes phenomenon [24] (which establishes the need for a reasonable balance between the number of training samples and the number of spectral bands in order to ensure a reliable classification [25], [26]), are important challenges to deploy the full potential of HSI technology with traditional pixel-based DNN approaches.

A significant evolution in DL techniques was the adaptation of biological visual cortex neurons into DNN architectures, with the implementation of convolutional neural networks (CNNs) [22]. Inspired by the local receptive field of such visual cortex neurons (activated or not in the presence of certain types of visual stimuli), CNN-based models rely on the application of a sliding n-dimensional kernel on the input data of each layer. This allows for the exploitation of the visual properties of an image, learning features at certain positions of such an image and applying these features as filters to the rest of the image in order to obtain a feature-activation map [27], [28]. In this sense, the contextualization provided by the spatial components $n_1 \times n_2$ of the HSI data cube $\mathbf{X}$ can greatly reduce the variability of spectral samples by interpreting the data surrounding the pixels as belonging to the same class, which reinforces the information contained in the target pixel, reducing also the well-known "salt and pepper" noise of spectral classifiers.

CNN models exhibit excellent performance in HSI data classification through the development of a wide range of architectures from traditional spectral-based ones (CNN1D) to spatial (CNN2D) and spectral–spatial (CNN3D) models. For instance, Hu *et al.* [29] implemented a five-layer CNN1D to classify HSI data in the spectral domain, and Yue *et al.* [30] developed a CNN3D to classify HSI data taking into account spectral–spatial information. Zhao and Du [31] exploited a CNN2D model as a highly confident spatial feature extractor. Chen *et al.* [32] reviewed CNN1D, CNN2D, and CNN3D models for deep HSI feature extraction (FE) and classification. In order to enhance the classification results, several improvements have been added to the CNN architecture. For instance, Yu *et al.* [33] implemented a three-layer CNN2D model with 1×1 kernels inspired by the network-in-network (NIN) model [34] in order to overcome the presence of highly correlated bands in the HSI data cube. He *et al.* [35] combined the information contained in HSI-extracted covariances with a CNN2D model. Paoletti *et al.* [36] presented a faster end-to-end CNN3D that improved the classification accuracy, taking into account the full spectral signatures contained in HSI data.

Despite the aforementioned results, CNN models still face certain limitations related to the intrinsic characteristics of HSI data and the (high) number of parameters and the depth of the network. In particular, CNNs need a large amount of training data to properly adjust their weights [37]. They also require some variability in the data in order to extract more features [28]. Although HSI data often exhibit a wide variety of samples, very limited labeled data are often available due to their high cost, which, in the end, hampers the FE process and leads to overadjustment (overfitting) in the convolutional model's parameters.

In addition, the implementation of very deep CNN models through the stacking of successive layers has proved to be inefficient itself [38], since a significant degradation can be observed in both the forward propagation of the data and the backpropagation of the gradient signal through the layers (vanishing gradient problem) [39], [40]. To overcome these issues, the residual learning aims to facilitate data reusability through identity functions implemented by skip or residual connections. Residual networks (ResNets) [38] and other residual-based architectures (such as highway networks [41], DenseNets [42], or ResNets of Resnets (RoRs) [43]) have emerged as the current state-of-the-art in image processing [44], allowing for the development of highly complex and deep architectures, using hundreds to thousands of layers [45]. These techniques, aimed at enhancing the propagation of data through the network, have been successfully adopted in several HSI classification works [46]–[48].

However, ResNets exhibit some shortcomings in terms of architecture optimization. In fact, residual-based models for HSI classification are quite sensitive to minor architectural changes, in particular, the selection of an appropriate kernel size has a significant impact on the final classification accuracy due to the low spatial resolution of HSI data cubes [47]. In contrast, at certain levels of depth, adding more or less layers to the network does not impact the classification result significantly [48]. In turn, this obviously affects the number of parameters that must be stored and trained. The understanding of the optimal number of parameters required by a certain architecture (the number of layers, kernel sizes, and so on) is quite critical, but it is often hand-crafted and adjusted by trial and error.

B. Rethinking the ResNet Model for HSI Classification

DNNs (in general) and ResNets (in particular) have been interpreted as a discrete sequence of L stacked layers, where each one applies its transformation to the input data until reaching a final classification decision, which is performed by the last layer. This implies that the ResNet model is evaluated at fixed intervals of "time," defined by the layer depth. Also, assuming that each layer has the same number of neurons n_{neurons} (which can be interpreted as the kernel's size in the convolutional architecture), the number of trainable parameters depends directly on L, so the complexity of the network (and its memory consumption) grows linearly with the $\mathcal{O}(L)$ order, which could have an impact on the model's overfitting. Under the same assumption, the computational time of the inference stage also depends on L.

The aforementioned implications provide an idea of the importance of the model's depth. As a result, the selection of L must be carefully done. In fact, the main goal of this paper is focused on two important aspects: 1) checking the effects of the depth when $L \to \infty$ and 2) analyzing strategies to provide the network with constant and low memory cost (in terms of the number of parameters). In this context, the FE function applied by each residual unit can be interpreted as the explicit Euler discretization of a continuous-time transformation [49], [50]. Following this interpretation, the entire ResNet model can be described through an ordinary differential equation (ODE) [51], [52], whose evaluation at different times will determine the model's solution [53], [54].

With the aforementioned ideas in mind, the main contribution of this article is to redefine the traditional architecture of the ResNet model (in the context of HSI data classification) by means of a continuous-time vision using ODEs, developing a residual-based DNN with a significantly reduced number of trainable parameters (thus effectively dealing with overfitting issues) and constant and low memory cost. These are important advantages in the area of HSI classification. More specifically, this article proposes, for the first time in the literature, the implementation of a continuous-depth ResNet with a parameterized spectral–spatial ODE in order to perform HSI data classification.

The remainder of this article is organized as follows. Section II introduces our newly developed model (called hereinafter ODEnet). Section III validates the newly proposed model by providing a detailed discussion of the results obtained using four widely used HSI data sets. Finally, Section IV concludes this article with some remarks and hints at plausible future research lines.

II. Methodology

A. Residual Units as Discrete Steps of Blocks

DNN architectures are stacks of L hidden blocks [55] F_1–F_L, where each one F_l is given by the following mapping function:

$$\mathbf{X}^{(l)} = F_l(\mathbf{X}^{(l-1)}, \mathbf{W}^{(l)}, b^{(l)}) \tag{1}$$

where $\mathbf{X}^{(l-1)}$ and $\mathbf{X}^{(l)}$ are the input and output data, respectively, and $\mathbf{W}^{(l)}$ and $b^{(l)}$ are the weights and biases of the lth mapping function F_l. In order to address the classification problem $f : \mathcal{X} \to \mathcal{Y}$, the DNN model assigns a classification map $\mathbf{Y} \in \mathbb{R}^{n_{\text{samples}}}$ to the given input $\mathbf{X} \in \mathbb{R}^{n_{\text{samples}} \times n_{\text{bands}}}$ by applying L sequential operations defined by (1). In this sense, the classification function $f(\cdot, \theta)$ can be reinterpreted as the concatenation of the processing at each layer processing as follows:

$$\mathbf{Y} = f(\mathbf{X}, \theta) = \hat{F}(F_L(F_{L-1}(\cdots F_1(\mathbf{X}) \cdots))) \tag{2}$$

where $\mathbf{X}$ is the original input data, $F_l(\cdot)$ is the mapping function performed by the lth network's block, and $\hat{F}(\cdot)$ is the final classification layer, while θ comprises the network's parameters [49]. In this regard, instead of considering the classification mapping as a global problem, the DNN model splits it into L mapping functions F_l, where the goal of the classification is to learn the parameters of each F_l that better minimize the convex loss function given by the following:

$$E = \frac{1}{n_{\text{samples}}} \sum_{i=1}^{n_{\text{samples}}} \| f(\mathbf{X}_i, \theta) - \mathbf{X}_i \|_2 . \tag{3}$$

If we focus on convolutional-based models, the data transformation defined by (1) is tailored in an FE stage defined by a *kernel operation* [36], which allows to easily combine the spatial–contextual information with the spectral information.

In this context, the CNN maintains the original 3-D data structure, adding a lot of flexibility to the model and a natural way to include the spectral–spatial information. Moreover, the internal structure of the CNN's layers and their operations (based on local receptive fields) have promoted it as a highly accurate feature extractor.

Two main parts can be observed in an end-to-end CNN classifier network: 1) the *FE stack*, which obtains high-level representations of the input data (also feature maps) and is usually comprised of a hierarchy of convolutional, nonlinear, and subsampling layers, among others and 2) the *FC classifier*, which actually labels the data from the previously obtained feature maps and is implemented as a standard MLP.

Focusing on the FE stack, it is usually adopted to implement an architecture of several hierarchically stacked extraction and detection stages, where the lth stage defines the lth mapping function F_l, following the notation of (1). Moreover, each F_l is usually comprised of: 1) the *convolutional layer*; 2) the *nonlinear layer*; 3) the *normalization layer*; and 4) the *pooling layer*, as (4) shows, although both the order and the type of layers may vary from one CNN architecture to another (even from one stage to another)

$$\mathbf{A}^{(l)} = (\mathbf{W}^{(l)} *_{k\times k\times q} \mathbf{X}^{(l-1)}) + b^{(l)} \tag{4a}$$

$$\hat{\mathbf{A}}^{(l)} = \frac{\mathbf{A}^{(l)} - \text{mean}[\mathbf{A}^{(l)}]}{\sqrt{\text{var}[\mathbf{A}^{(l)}] + \epsilon}} \cdot \gamma + \beta \tag{4b}$$

$$\tilde{\mathbf{A}}^{(l)} = \mathcal{H}(\hat{\mathbf{A}}^{(l)}) \tag{4c}$$

$$\mathbf{X}^{(l)} = \mathcal{P}_{k\times k}(\tilde{\mathbf{A}}^{(l)}). \tag{4d}$$

The convolutional layer performs the basic FE task of the model. The spectral–spatial convolutional layer of the F_l mapping function is comprised of a group of K filters with $\mathbf{W}^{(l)} \in \mathbb{R}^{k\times k\times q}$ weights and $b^{(l)}$ biases, being $k \times k \times q$ the local receptive field of the layer. In consequence, each layer creates a linear kernel that slides (following a stride s) and overlaps the input data, convolving ($*$) its filters on local patches of the data, as (4a) indicates. As a result, the obtained output volume is comprised of K feature maps.

After the convolutional layer, it is common to include a batch normalization layer, which imposes a Gaussian distribution on the obtained feature maps with the aim of preventing the data degradation and vanishing gradient problems (mainly due to the covariance shift that the data suffers). Equation (4b) gives the regularization expression, where ϵ is a parameter that allows a certain numerical stability and γ and β are learnable parameters.

Following the normalization layer, a nonlinear layer $\mathcal{H}(\cdot)$ defined by (4c) is introduced in order to extract the activation maps from the convolutional output volume. In fact, this layer embeds a nonlinear activation function, which encodes the detector stage of the network [56], learning the nonlinear representations and relationships inside the data. Many activation functions can be selected, such as the tanh, sigmoid, or rectified linear unit (ReLU) [57], which allows a faster training of the model due to its high computational efficiency.

Finally, the extraction and detection stage ends with the pooling layer $\mathcal{P}_{k\times k}(\cdot)$ given by (4d), which performs a downsampling strategy with the aim of reducing the spatial dimensions of the output volume by applying, for instance, a max, average, or sum operation on the spatial receptive field of dimensions $k \times k$.

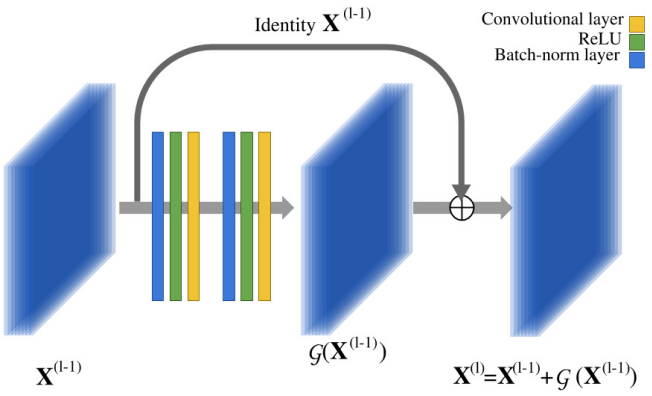


Fig. 1. Graphical representation of the lth residual unit architecture, F^l, comprised of two FE and detection stages. Each stage is comprised of normalization, nonlinear, and convolutional layers. The application of these stages gives, as a result, the output volume $\mathcal{G}(\mathbf{X}^{(l-1)})$, to which an identity mapping is added at the end of the residual unit, obtaining the final residual output volume $\mathbf{X}^{(l)} = \mathbf{X}^{(l-1)} + \mathcal{G}(\mathbf{X}^{(l-1)})$.

Based on the CNN architecture, the success of the ResNet model lies in the skip and residual connections, in which grouped operation layers (i.e., convolutional, pooling, and normalizing layers) and nonlinear activation functions comprise of the basic blocks for data mapping [47], as shown in Fig. 1. These residual units allow for the development of deeper architectures, where the inputs and outputs of each unit are connected through a residual connection, performing an additional identity mapping that allows to propagate the information from previous blocks to the rest of the network. In this context, for the lth residual unit, the FE and detection stages given by (4) can be reformulated as follows:

$$\mathbf{A}^{(l)} = \mathbf{X}^{(l-1)} + \mathcal{G}(\mathcal{W}^{(l)}, \mathbf{X}^{(l-1)}, \mathcal{B}^{(l)}) \tag{5a}$$

$$\mathbf{X}^{(l)} = \mathcal{H}(\mathbf{A}^{(l)}) \tag{5b}$$

where $\mathcal{G}(\cdot)$ comprises all the operations applied over the residual unit's input data, i.e., all the convolutions, poolings, normalizations, and activations applied over $\mathbf{X}^{(l)}$, being $\mathcal{W}^{(l)}$ and $\mathcal{B}^{(l)}$ the weights and biases of the layers involved in the residual block, respectively. Moreover, the additive residual mapping function added to $\mathcal{G}(\cdot)$ allows to recycle the features obtained at the previous level of abstraction.

Following (2), the ResNet defines each mapping function F_l through (5). In this context, the neural model can be interpreted as a discrete sequence of L hidden units or mapping functions, dividing the classification process into L steps, so that each F_l defines a hidden *state* of the process, which becomes more manageable, with simple and detailed steps that allow for a more accurate final classification. However, this implies that the quality of the model depends on its trainable parameters, and the number of trainable parameters depends directly on L. This has two main implications. On the one hand, the memory consumption grows linearly [with $\mathcal{O}(L)$ order] and there is an increment of training data that the model must assume in order to properly learn the network's parameters, avoiding the

overfitting problem. On the other hand, although the residual connections alleviate the aforementioned problem, each new unit that is added to the model introduces a small error [38], [58], which may hinder the model's overall performance. These issues are particularly critical when dealing with highly variable HSI data sets.

B. Residual Units as Discrete Steps of ODEs

Our goal is to develop a residual model with constant and low memory cost through a significant reduction of the number of trainable parameters. We follow the premise of traditional optimization models: solving a lot of small steps is often better than solving fewer and more complex ones [50]. In this sense and following (2), we propose to implement an ResNet model for HSI data classification in which the forward problem is comprised of infinitesimal steps, i.e., $L \rightarrow \infty$ [54]. Each of these steps performs (5), which describes an explicit Euler discretization step of the ODE [51], [52]. Below, the mathematical relationship between ResNet models and ODEs is described in detail.

We focus on the first-order ODE expressions. Following Euler's solving method, any first-order ODE can be expressed as an initial value problem (IVP) of the form:

$$\frac{\mathrm{d}\mathbf{z}(t)}{\mathrm{d}t} = f(t, \mathbf{z}(t), \theta), \text{ with } \mathbf{z}(t_0) = \mathbf{z}_0 \tag{6}$$

where t_i is an independent variable defined in terms of *time* in an observation interval $\{0, ..., T\}$, $f(\mathbf{z}(t), t, \theta)$ is a known and continuous function with parameter θ, and $\mathbf{z}(t)$ is the unknown function that must be approximated, with initial state $\mathbf{z}_0$ at time t_0. In fact, the goal of any ODE function is to recover the closest and most accurate value $\mathbf{z}_i$ of the unknown function $\mathbf{z}(t_i)$ at each observation point t_i.

From a geometric point of view, knowing $\mathbf{z}(t_0) = \mathbf{z}_0$, an approximation of $\mathbf{z}(t_i) = \mathbf{z}_i$ in any step t_i can be performed by drawing the tangent line from previous-known points as follows:

$$\mathbf{z}_1 \approx \mathbf{z}_0 + f(t_0, \mathbf{z}_0, \theta)(t_1 - t_0) \tag{7a}$$

$$\mathbf{z}_i \approx \mathbf{z}_{i-1} + f(t_{i-1}, \mathbf{z}_{i-1}, \theta)(t_i - t_{i-1}) \tag{7b}$$

$$\mathbf{z}_T \approx \mathbf{z}_{T-1} + f(t_{T-1}, \mathbf{z}_{T-1}, \theta)(t_T - t_{T-1}). \tag{7c}$$

Generalizing the discrete steps defined above, it can be stated that any $\mathbf{z}(t_i)$ can be approximated by (7b). Assuming that the ith observation point is connected to the first one (following the relation $t_i = t_0 + \alpha \cdot i$, where α is a step-size), the Euler discretization method claims that each point t_i is related to the immediately preceding one, t_{i-1}, through the step-size α as follows: $t_i = t_{i-1} + \alpha$. Including this relationship in (7b), Euler's method gives a solution for $\mathbf{z}(t_i)$ as

$$\mathbf{z}_i = \mathbf{z}_{i-1} + f(t_{i-1}, \mathbf{z}_{i-1}, \theta) \cdot (\cancel{t_{i-1}} + \alpha - \cancel{t_{i-1}}) \tag{8a}$$

$$\mathbf{z}_i = \mathbf{z}_{i-1} + \alpha \cdot f(t_{i-1}, \mathbf{z}_{i-1}, \theta). \tag{8b}$$

At this point, it is easy to observe the relationship between the ResNet model and the first-order ODE. Focusing on (5), we can simplify it into a more condensed form

$$\mathbf{X}^{(l)} = \mathbf{X}^{(l-1)} + \mathcal{G}(\mathbf{X}^{(l-1)}, \theta_l). \tag{9}$$

The similarities between (8b) and (9) are evident. In fact, (9) defines an explicit Euler discretization step of the first-order ODE, where the step size is set to $\alpha = 1$ and the known function is implemented by the extraction and detection stages $\mathcal{G}(\cdot)$ of the residual unit, being parameterized by the weights and biases of the layers that comprise the residual unit $\theta_l = (\mathcal{W}^{(l)}, \mathcal{B}^{(l)})$. In other words, the ODE function is, in fact, a CNN.

Following this intuition, we can replace the discrete block-by-block performance of a ResNet model by a continuous-time ODE function. In particular, we assume a residual model with $L \rightarrow \infty$ equal residual units. In this sense, each mapping function F_l has to perform the same extraction and detection stages in $\mathcal{G}(\cdot)$, so each unit has the same number of parameters θ and works in the same feature space $F_1, \ldots, F_L \in \mathbb{R}^{\hat{n}_1 \times \hat{n}_2 \times \hat{n}_3}$, where $\hat{n}_1 \times \hat{n}_2 \times \hat{n}_3$ are the spatial–spectral dimensions of the feature maps.

In this way, the successive transformations given by (2), $F_1, \ldots, F_L$, can be interpreted as the continuous mapping function $F(t)$ evaluated at different times (with a relationship between layers and time). So, at the ith observation time, we can obtain $F(t_i) = \mathbf{X}_i$. As a result, the residual model can be reformulated as the ODE in (10b), which gives the discretization step of Euler's method and the expression of the first-order ODE

$$\mathbf{X}_i = \mathbf{X}_{i-1} + \mathcal{G}(t_{i-1}, \mathbf{X}_{i-1}, \theta) \tag{10a}$$

where

$$\frac{\mathrm{d}F(t)}{\mathrm{d}t} = \mathcal{G}(t, F(t), \theta), \text{ with } F(t_0) = \mathbf{X}_0. \tag{10b}$$

As it can be observed, the ODE is implemented by the neural network defined by $\mathcal{G}(\cdot)$ and parameterized by θ.

C. Proposed ODEnet for HSI Classification

We propose, for the first time in the literature, to reinterpret the ResNet model (for HSI data classification) as a continuous transformation given by the first-order ODE described in (10b). Fig. 2 gives a general overview of the proposed ODEnet, which receives as input the HSI data cube with dimensions $\mathbf{X} \in \mathbb{R}^{d \times d \times n_{\text{bands}}}$. In fact, the model is fed with hyperspectral patches cropped from the original HSI cube, comprised of $d \times d$ pixels and n_{bands} spectral bands, where the label corresponds to the central pixel of the patch. Also, in order to take advantage of border pixels, a mechanism for mirroring the image edges has been implemented [36].

The proposed network architecture is divided into the FE layers and the final classification layers. Focusing on the FE layers, they are grouped into three categories: 1) FE head; 2) FE body; and 3) FE tail. The FE head performs a downsampling of the data, reducing noise, and cleaning the information contained in the input. It is comprised of a convolutional layer F_1 and a residual unit F_2. F_1 prepares the input data, extracting the initial features from the HSI cube, which are fundamental to the performance of the rest of the layers. During the training process, these features will become more and more robust and discriminative, being decisive for the final classification. F_2 has been implemented following the *preactivation* architecture

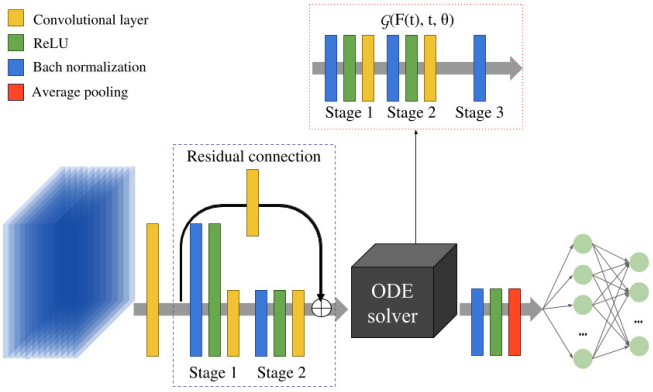


Fig. 2. Architecture of the proposed ODEnet for HSI data classification. The feature extractor part is comprised of three well-differentiated parts: 1) a preprocessing step that filters the spatial–spectral noise and extracts low-level feature representations; 2) the ODE solver that evaluates the function $\mathcal{G}(\cdot)$ defined by a neural network; and 3) the ODE solver's output, after being refined, is employed to perform the final classification, implemented by two FC layers.

proposed in [45], performing data downsampling, and it is comprised of two FE and detection stages with normalization, nonlinear, and convolutional layers, adding a convolutional layer on the skip connection to maintain the data shape.

The obtained features are sent to the FE body, which is implemented by a continuous-time ResNet. In this context, the ODE implemented by (10b) has been parameterized by a CNN model. As Fig. 2 shows, this model follows the *preactivation* architecture [45] and has three stages, where each one is comprised of normalization, nonlinear, and convolutional layers (stages 1 and 2), and a normalization layer (stage 3). This ODE is solved from some initial time t_0 to some ending time t_T, creating an integration time interval $[0, T]$. Furthermore, during each forward pass, the traditional discrete-layer execution of the model is eventually replaced by $\hat{L}$ evaluations of (10b), performed by a black-box solver in the interval $[0, T]$, which receives as the initial condition $\mathbf{X}_0$ the output of F_2, the known function $\mathcal{G}(\cdot)$ and its parameters θ, in addition to the integration time interval, and a tolerance threshold of the estimated error, tol

$$F(t_T) = \mathbf{X}_T = \text{ODEsolver}(\mathbf{X}_0, \mathcal{G}, \theta, [t_0, t_T], \text{tol}). \quad (11)$$

Equation (11) can be performed by any off-the-shelf ODE solver. There is a great variety of methods for this purpose, grouped in different categories depending on their internal characteristics and working modes [59], being some of the methods framed within the most well-known Runge–Kutta family, which are as follows.

1) *Forward Euler:* This is the most popular numerical explicit method for solving the first-order ODEs. It is also the simplest method to implement, where the new states are obtained through previously known ones by the intersection of tangent lines, as (8) shows. Given the first-order ODE of (6) and using α as the step size, the approximation error of Euler's discretization method will be proportional to $\mathcal{O}(\alpha^2)$.
2) *Explicit Midpoint Method also Known as the Modified Euler method:* Given (6), the evaluations are made at $\alpha/2$, so this method determines the value $\mathbf{z}(t_i) = \mathbf{z}_i$ as the following approximation:

$$\mathbf{z}_i = \mathbf{z}_{i-1} + \alpha \cdot f\left(t_{i-1} + \frac{\alpha}{2}, \mathbf{z}_{i-1} + \frac{\alpha}{2} \cdot k_1\right) \quad (13a)$$

$$k_1 = f(t_{i-1}, \mathbf{z}_{i-1}) \quad (13b)$$

This method reduces the estimation error when Euler's step size is too high and the tangent needs to be elongated to find the intersection point.
3) *Fourth-Order Runge–Kutta (RK4) Method:* This is the most widely used method of the Runge–Kutta family. Inspired by the midpoint method, the basic idea is that, given two equidistant points $t_i = t_{i-1} + \alpha$, the function $\mathbf{z}(t_i) = \mathbf{z}_i$ can be approximated as the sum of the previously known value and the weighted average of s slopes [60]

$$\mathbf{z}_i = \mathbf{z}_{i-1} + \sum_{n=1}^{s} b_n, k_n \quad (14a)$$

$$k_1 = \alpha f(t_{i-1}, \mathbf{z}_{i-1}) \quad (14b)$$

$$k_n = \alpha f\left(t_{i-1} + c_n\alpha, \mathbf{z}_{i-1} + \sum_{\hat{n}=1}^{n-1} a_{n,\hat{n}} k_{\hat{n}}\right) \quad (14c)$$

where $a_{n,\hat{n}}$, b_n, and c_n are weighted coefficients. In this sense, given (6), the RK4 method determines the value at t_i as an approximation of the previously known $\mathbf{z}_{i-1}$ and the weighted average of four increments: $(k_1+2k_2+2k_3+k_4)/6$, which are calculated on certain points of the slope defined by $f(\mathbf{z}(t), t, \theta)$, in particular, the starting, ending, and midpoints [61]

$$\mathbf{z}_i = \mathbf{z}_{i-1} \cdot \frac{1}{6}(k_1 + 2 \cdot k_2 + 2 \cdot k_3 + k_4) \quad (15a)$$

$$k_1 = \alpha f(t_{i-1}, \mathbf{z}_{i-1}) \quad (15b)$$

$$k_2 = \alpha f\left(t_{i-1} + \frac{\alpha}{2}, \mathbf{z}_{i-1} + \frac{k_1}{2}\right) \quad (15c)$$

$$k_3 = \alpha f\left(t_{i-1} + \frac{\alpha}{2}, \mathbf{z}_{i-1} + \frac{k_2}{2}\right) \quad (15d)$$

$$k_4 = \alpha f\left(t_{i-1} + \alpha, \mathbf{z}_{i-1} + k_3\right). \quad (15e)$$

Following (15), the approximation error is proportional to $\mathcal{O}(\alpha^4)$, being more precise than the two previous methods.
4) *Dormand-Prince Method (DOPRI5):* This is an explicit and adaptive Runge–Kutta method to calculate the fourth- and fifth-order solutions. In fact, following (14), it calculates seven slopes: k_1–k_7, which are employed to calculate two approximations of $\mathbf{z}(t_i) = \mathbf{z}_i$ by two different linear combinations. Equation (12a), as shown at the bottom of the next page, gives the first approximation, with $\mathcal{O}(\alpha^4)$ order, while (12b) gives the second approximation, with $\mathcal{O}(\alpha^5)$ order. An interesting aspect of the DOPRI5 solver is its ability to adapt the step size α to keep the estimated error $|\hat{\mathbf{z}}_i - \mathbf{z}_i|$ below a predefined threshold. The updating of the optimal step size α_{opt} is obtained as

$$s = \left(\frac{\text{tol} \cdot \alpha}{2|\hat{\mathbf{z}}_i - \mathbf{z}_i|}\right)^{\frac{1}{5}} \quad (16a)$$

$$\alpha_{\text{opt}} = s \cdot \alpha \quad (16b)$$

TABLE I
PROPOSED NETWORK TOPOLOGY

Feature extraction network								
	Module ID	**Sub-module**	**Norm.**	**Activation**	**Kernel**	**Stride**	**Padding**	**Pooling**
FE-head	F_1	-	-	-	$64 \times 3 \times 3 \times n_3$	1	-	-
	F_2	Stage 1	Yes (64)	ReLU	$64 \times 3 \times 3 \times 64$	2	Yes	-
		Stage 2	Yes (64)	ReLU	$64 \times 3 \times 3 \times 64$	1	-	-
		Skip connection	-	-	$64 \times 1 \times 1 \times 64$	2	-	-
FE-body	$\mathcal{G}(\cdot)$	Stage 1	Yes (64)	ReLU	$64 \times 3 \times 3 \times 64$	1	Yes	-
		Stage 2	Yes (64)	ReLU	$64 \times 3 \times 3 \times 64$	1	Yes	-
		Stage 3	Yes (64)	-	-	-	-	-
FE-tail	F_3	Stage 1	Yes (64)	ReLU	-	-	-	Average (1×1)

Classification network			
	Layer	**Neurons**	**Activation**
MLP	F_4	64	ReLU
	F_5	$n_{classes}$	Softmax

where tol defines the tolerance level, which provides robustness and reliability to the model.

In addition to obtaining the corresponding state $\mathbf{X}_T = F(t_T)$ at t_T (forward-propagation), the ODEsolver should optimize the network's parameters associated with the differential equation $\mathcal{G}(t, F(t), \theta)$ by backpropagating the internal error signal $E_{\text{ode}}(\cdot)$ defined by the following expression:

$$E_{\text{ode}}(F(t_T)) = E\left(F(t_0) + \int_{t_0}^{t_T} \mathcal{G}(t, F(t), \theta)dt\right). \quad (17)$$

This optimization can be implemented by two methods: 1) traditional integration through a *Runge–Kutta integrator*, for instance, or 2) employing the *adjoint method* [54], [62]. The first one directly integrates the operations of the forward pass and still presents an important memory requirement in the sense that, for $\hat{L}$ evaluations, the memory cost grows to the order of $\mathcal{O}(\hat{L})$. However, the adjoint method allows to optimize the parameters of $\mathcal{G}(\cdot)$ while significantly reducing their management, keeping constant the memory cost in the order $\mathcal{O}(1)$ [54].

Finally, the FE-layers end with the FE tail, which receives $\mathbf{X}_T$, the estimated output of the ODEsolver at evaluation time t_T, and performs a final processing. This entails an FE and detection stages, denoted as F_3, which comprises normalization, nonlinear, and average pooling layers. The obtained feature maps are then reshaped and sent to the classifier, which has been implemented as an MLP with two FC layers: F_4 and F_5, where the last one produces the final classification.

Table I gives the topology details of the proposed ODEnet. Moreover, our ODEnet model has been trained by the stochastic gradient descend (SGD) optimizer to minimize the classification loss given by (3), with input patches of 11×11, using 160 epochs and 0.1 as the learning rate, taking into account a momentum of 0.9 and learning rate decay, and a batch size of 128, while the ODEsolver is implemented via the DOPRI5 solver with a tolerance fixed to $\text{tol} = 1e-3$ and an integration time interval of $[0, 1]$, which directly controls the number of evaluations $\hat{L}$ of the model by obtaining the optimal step size α.

$$\mathbf{z}_i = \mathbf{z}_{i-1} + \frac{35k_1}{384} + \cancel{0k_2} + \frac{500k_3}{1113}\frac{125k_4}{192} - \frac{2187k_5}{6784} + \frac{11k_6}{84} + \cancel{0k_7} \quad (12a)$$

$$\hat{\mathbf{z}}_i = \mathbf{z}_{i-1} + \frac{5179k_1}{57600} + \cancel{0k_2} + \frac{7571k_3}{16695}\frac{393k_4}{640} - \frac{92097k_5}{339200} + \frac{187k_6}{2100} + \frac{k_7}{40} \quad (12b)$$

$$k_1 = \alpha f(t_{i-1}, \mathbf{z}_{i-1}) \quad (12c)$$

$$k_2 = \alpha f\left(t_{i-1} + \frac{\alpha}{5}, \mathbf{z}_{i-1} + \frac{k_1}{5}\right) \quad (12d)$$

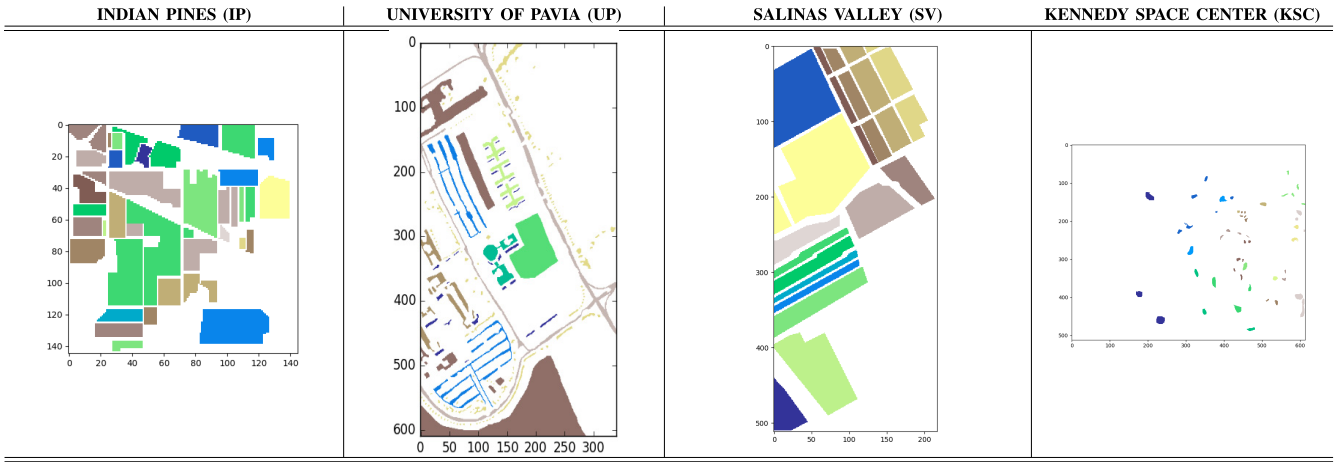
$$k_3 = \alpha f\left(t_{i-1} + \frac{3\alpha}{10}, \mathbf{z}_{i-1} + \frac{3k_1}{40} + \frac{9k_2}{40}\right) \quad (12e)$$

$$k_4 = \alpha f\left(t_{i-1} + \frac{4\alpha}{5}, \mathbf{z}_{i-1} + \frac{44k_1}{45} - \frac{56k_2}{15} + \frac{32k_3}{9}\right) \quad (12f)$$

$$k_5 = \alpha f\left(t_{i-1} + \frac{8\alpha}{9}, \mathbf{z}_{i-1} + \frac{19372k_1}{6561} - \frac{25360k_2}{2187} + \frac{64448k_3}{6561} - \frac{212k_4}{729}\right) \quad (12g)$$

$$k_6 = \alpha f\left(t_{i-1} + \alpha, \mathbf{z}_{i-1} + \frac{9017k_1}{3168} - \frac{355k_2}{33} - \frac{46732k_3}{5247} + \frac{49k_4}{176} - \frac{5103k_5}{18656}\right) \quad (12h)$$

$$k_7 = \alpha f\left(t_{i-1} + \alpha, \mathbf{z}_{i-1} + \frac{35k_1}{384} + \cancel{0k_2} + \frac{500k_3}{1113}\frac{125k_4}{192} - \frac{2187k_5}{6784} + \frac{11k_6}{84}\right) \quad (12i)$$



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

Fig. 3. Number of available labeled samples in the IP, UP, SV, and KSC HSI data sets.

III. Experimental Results

A. Experimental Environment

In order to study the performance of the proposed ODEnet for HSI classification, an implementation has been developed and tested on a hardware environment with a sixth-generation Intel Core i7-6700 K processor with 8M of Cache and up to 4.20 GHz (four cores/eight-way multitask processing), installed over an ASUS Z170 pro-gaming motherboard. The available memory is 40 GB of DDR4 RAM with serial speed of 2400 MHz and a Toshiba DT01ACA HDD with 7200 RPM and 2 TB of storage capacity. Also, a graphic processing unit (GPU) NVIDIA GeForce GTX 1080 with 8-GB GDDR5X of video memory and 10 Gb/s of memory frequency is available. In order to provide an efficient implementation, the proposed model has been parallelized over the GPU using CUDA 9.0 and cuDNN 7.1.1 language over the Pytorch framework, with Ubuntu 18.04.1 $\times$ 64 as the operating system.

B. Hyperspectral Data Sets

Fig. 3 presents the four real HSI data sets that have been considered in our experiments: Indian Pines (IP), Salinas Valley (SV), and Kennedy Space Center (KSC) scenes, acquired by the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) sensor [5], and the University of Pavia (UP) scene, captured by the Reflective Optics System Imaging Spectrometer (ROSIS) sensor [63]. A detailed description of these images is provided in the following.

1) The IP scene comprises an area with different agricultural fields in Northwestern Indiana, USA, imaged during a flying campaign of the AVIRIS sensor in 1992. The scene contains 145 $\times$ 145 samples, where each one comprises 20 m, and the spectral information consists of 200 bands in the wavelength range from 0.4 to 2.5 μm, after removing 24 noisy and corrupted bands. As it can be observed in Fig. 3, the ground truth of the IP scene contains a total of 16 different classes.
2) The UP image was acquired in 2001 by the ROSIS sensor over the UP, Northern Italy, capturing an urban area of 610 $\times$ 340 pixels, where each one comprises 1.3 m, and with spectral (103 bands, after elimination of noisy and corrupted bands) in the wavelength range from 0.43 to 0.86 μm. The number of different classes contained in the UP scene is nine.
3) The SV image was captured during a flying campaign of the AVIRIS sensor in 1998 over the agricultural area described as SV in CA, USA. The data comprise 512 $\times$ 217 pixels with the spatial resolution of 3.7 m/pixel and 200 spectral bands in the range from 0.4 to 2.5 μm (200 bands, after elimination of the noisiest bands). The available ground truth for the SV scene contains 16 classes.

4) Finally, the KSC scene was also gathered by the AVIRIS instrument in 1996 over the KSC in FL, USA. In this scene, 512 × 614 pixels were obtained with the spatial resolution of 20 m/pixel. The data comprises 176 spectral bands in the range from 0.4 to 2.5 μm after the removal of noisy bands. The available ground truth for this scene comprises 13 different classes.

C. Experimental Setting

To evaluate the classification performance of the proposed ODEnet for HSI classification, three widely used quantitative metrics have been considered: the overall accuracy (OA), average accuracy (AA), and Kappa coefficient. Moreover, the number of model's parameters and execution times has also been measured to determine the volume of data to be trained and the computational cost. In this regard, with the aim of providing a complete and detailed experimentation, several experiments have been carried out, which are as follows.

1) Our first experiment evaluates the performance of the proposed ODEnet by implementing it with different ODEsolvers, in particular, forward Euler (EULER), explicit midpoint (MIDPOINT), RK4, and DOPRI5. For this experiment, the IP data set has been considered, selecting randomly 10% of the available labeled samples for training and using the remaining 90% of the samples for testing, setting the tolerance threshold to tol $= 1e{-}3$. Each experiment has been executed ten times, and the average and standard deviations have been reported.
2) Our second experiment focuses on the DOPRI5 solver due to its ability to adapt the step size α, adapting, in turn, the number of evaluations $\hat{L}$ contained in the defined integration time interval $[t_0, t_T]$ to the complexity of the function, as opposed to the EULER, MIDPOINT, and RK4 methods that set a fixed size for α, making the same number of evaluations in each step. In this regard, our second experiment analyzes the behavior of the DOPRI5 solver with different tolerance thresholds, in particular: tol $= \{1e-1, 1e-2, 1e-3$ $1e-4, 1e-5\}$. For this purpose, the OA values, the number of evaluations during the forward and backward steps, and the training execution times have been measured. Again, in this experiment, we randomly select 10% of the available labeled samples of the IP data set for training and use the remaining 90% for testing. Each experiment is executed ten times and the average and standard deviations are reported.
3) Once the model's behavior has been evaluated with different solvers and tolerance levels, our third experiment performs several comparisons between the proposed ODEnet and the traditional ResNet model for spectral–spatial HSI data classification. In this context, this experiment compares the robustness of the models, analyzing their performance based on the amount of available training data, the number of parameters used by each model, and the evolution of the accuracy in each epoch. For a fair comparison, the ResNet has been implemented in the same way as the ODEnet, using the topology in Table I, and changing the ODEsolver by six residual units comprised of exactly the same stages as the proposed ODEnet's FE body, but adding the corresponding residual connections. Moreover, the proposed ODEnet has been implemented with the DOPRI5 solver, employing Runge–Kutta integration and adjoint methods and a tolerance threshold of $1e-3$. These models have been tested with all the available scenes. For the IP and KSC scenes, we have randomly selected 5%, 10%, and 15% of the available labeled samples for training and used the remaining samples for testing. The fact that we consider larger training percentages for these two images is due to the low spatial resolution and highly mixed nature of these scenes, which exhibit high intraclass variability. In turn, for the UP and SV scenes (which exhibit much larger spatial resolution), we have randomly selected 1%, 5%, and 10% of the available labeled samples for training, using the remaining samples for testing. In all the cases, we have executed each experiment ten times and the average and standard deviations are reported.
4) The fourth experiment compares the behavior of the proposed ODEnet models and the ResNet depending on different network configurations, in particular, the spatial windows' size of the network's input data and the depth of the convolutional filters. In this sense, the proposed models have been implemented with DOPRI5 during the forward pass, while employing both Runge–Kutta integrator and the adjoint method during the backward step. For each experiment, the 10% of IP and KSC and the 5% of UP and SV data sets have been considered to perform the training of the models.
   Regarding the first experiment, it compares the performance of the neural models when different amounts of spatial information confirm the network's input data. In this context, different window sizes have been considered, in particular input patches of 5 × 5, 7 × 7, 9 × 9, 11 × 11, 13 × 13, and 15 × 15 pixels have been tested. Separately, the second experiment compares the networks' behavior when the number of convolutional filters grows. In this regard, convolutional layers have been implemented with 8, 16, 32, 64, and 128 filters.
5) Our last experiment conducts a comparison of the proposed ODEnet with other widely used HSI classifiers. In this context, eight different classification methods have been selected to conduct the experimental validation. Specifically, three pixelwise classifiers (MLR, SVM with radial basis function kernel, and MLP), one deep spatial classifier (CNN2D), and three spectral–spatial deep architectures (CNN3D, ResNet, and the proposed ODEnet) have been considered. In this experiment, we have randomly selected 15% of the available labeled data from the IP and KSC scenes and used the remaining 85% of the labeled data for testing. Considering the higher spatial resolution of the UP and SV scenes, we have randomly selected 10% of the available labeled samples for these scenes and used the remaining 90% for testing. As in the previous experiments, we repeated

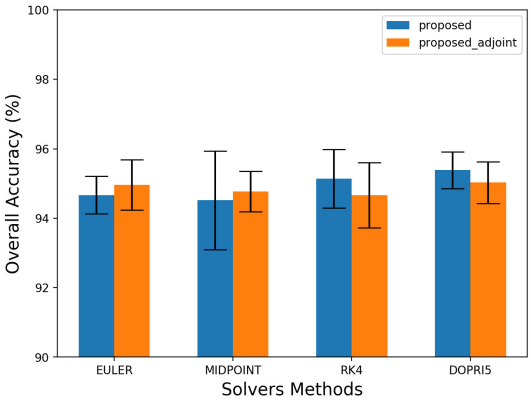


Fig. 4. OA values (and corresponding standard deviations) obtained by the proposed method (implemented with four different solvers with Runge–Kutta integration and adjoint methods) for the IP scene.

each experiment ten times and report the average and standard deviations. Moreover, for the spatial (CNN2D) and the spectral–spatial (CNN3D, ResNet, and ODEnet) methods, the original HSI scene has been cropped into patches of 11×11. In the case of the CNN2D, principal component analysis (PCA) has been used to reduce the number of spectral bands to a single principal component. All the hyperparameters of the considered methods have been optimally fixed to obtain the best possible performance for each method.

D. Experiment 1: Testing Different ODEsolvers

The performance of the proposed ODEnet depends on two main aspects: 1) the solver that performs the forward evaluation and 2) the backpropagation method that implements the reverse-mode differentiation. In this experiment, the fixed-α solvers: EULER, MIDPOINT, and RK4, and the adaptive solver: DOPRI5 have been compared using the IP data set, testing each one with Runge–Kutta integration (simply referred to ODEnet hereinafter) and the adjoint method (ODEnetAdj hereinafter).

Fig. 4 gives the obtained OA results and the standard deviations for each considered model. As a general comment, it should be noted that all methods achieve an OA greater than 94% with small differences between them. Specifically, the difference between the implementation of each solver with Runge–Kutta integration and adjoint method is very small, achieving very similar results.

If we compare the fixed-α solvers (EULER, MIDPOINT, and RK4) with the adaptive DOPRI5 solver, it can be observed that DOPRI5 reaches the best OA values for both backpropagation methods, Runge–Kutta integration, and adjoint, exceeding 95% OA with very low standard deviation, due to its capability of adapting the evaluations to the problem's complexity. Furthermore, MIDPOINT and RK4 exhibit the worse OA scores when implemented using Runge–Kutta integration and adjoint methods, respectively. In particular, the MIDPOINT method implemented with Runge–Kutta integration exhibits the highest standard deviation, because the adopted approximation strategy performed by calculating the midpoint of the slope is not the most appropriate for complex data such as HSI scenes.

E. Experiment 2: Testing Different Tolerance Thresholds for DOPRI5 Solver

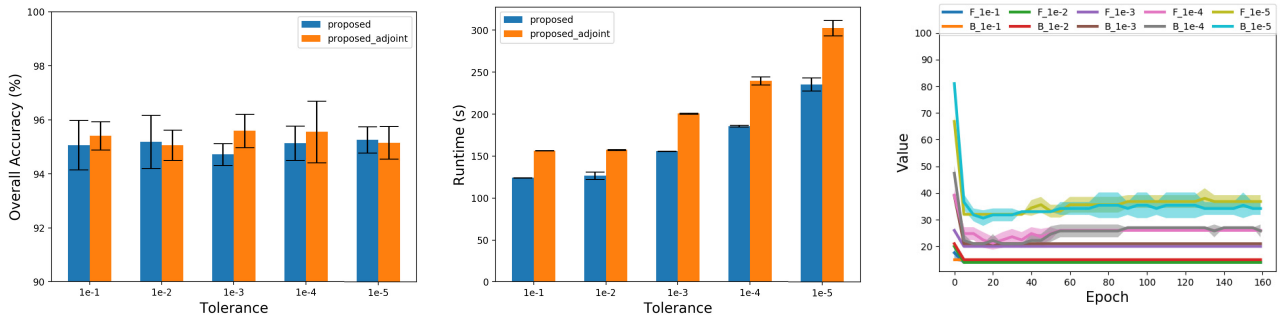
The DOPRI5 solver is able to adapt the step size α that controls the number of evaluation points ($\hat{L}$) carried out inside the integration time interval $[t_0, t_T]$, providing a flexible mechanism to adapt the ODE resolution to the complexity of the considered HSI data. In this sense, five different values for the tolerance threshold have been considered: $\{1e-1, 1e-2, 1e-3\ 1e-4, 1e-5\}$.

Fig. 5 shows the obtained results, comparing the obtained OA values [see Fig. 5(a)], the training runtimes [see Fig. 5(b)], and the number of evaluations performed during the forward and backward steps (for each tolerance value) [see Fig. 5(c)]. If we focus on Fig. 5(a), it can be observed that the tolerance threshold does not have a relevant impact on the OA values in the sense that the differences are very small and the slight variations are mainly due to the random procedure used for the selection of training samples.

However, if we focus on Fig. 5(b), it can be clearly observed that, for lower tolerances, the execution times gradually increase, being the implementations with DOPRI5 and adjoint method the slowest ones. This is due to the number of evaluations $\hat{L}$ that need to be carried out both in the forward evaluations and the backward propagation. To further investigate this issue, Fig. 5(c) focuses on the DOPRI5 solver implementation with the adjoint method. In general, the number of forward and backward evaluations, in this case, is high in the early epochs with the aim of adjusting them to minimize the approximation error, descending abruptly until the number becomes stable in subsequent epochs. In addition, for lower tolerances, it can be observed that the number of evaluations is higher than the tolerance values of $1e-1$ and $1e-2$, where the difference is minimal. With the aforementioned observations in mind, we consider a tolerance of $1e-3$ as a good choice, in the sense that it provides a good balance between performance and training times, together with a sufficiently high number of evaluations.

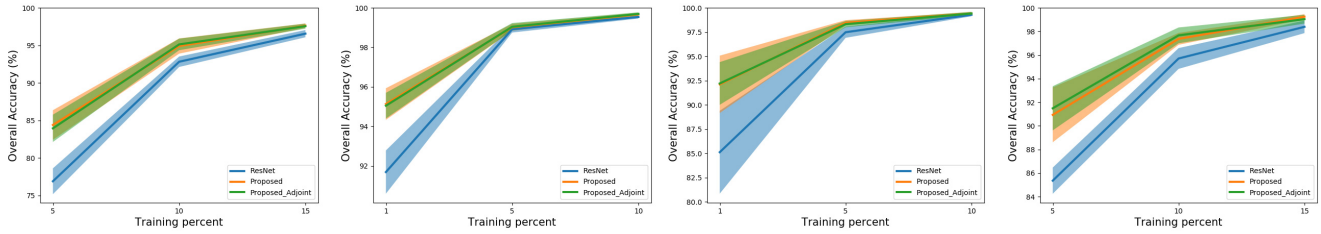
F. Experiment 3: Comparing ODEnet With ResNet

In this experiment, we illustrate the benefit of implementing a ResNet-inspired model as a continuous function defined by an ODE. Fig. 6 shows the OA evolution of the proposed ODEnet when different amounts of training samples are available. In general, the proposed method, implemented either with DOPRI5 and Runge–Kutta integrator (ODEnet) or with the adjoint method (ODEnetAdj), exhibits the best OA results for all the considered HSI scenes, regardless of the training percentage employed. The differences between our ODEnet/ODEnetAdj models and the ResNet become particularly evident when a very few training samples are available with the proposed models exhibiting the most robust results. Again, the observable differences between the Runge–Kutta integrator and the adjoint method are quite small, being the



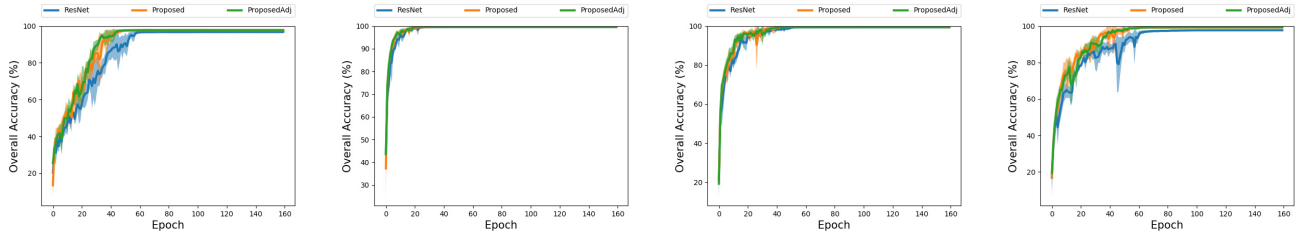
(a) (b) (c)

Fig. 5. Performance of ODEnet (on the IP scene) with the DOPRI5 solver, using Runge–Kutta integration and adjoint methods, considering different tolerance values. Specifically, we analyze the impact on (a) OA, (b) training runtimes, and (c) number of evaluations per epoch during the forward and backward steps of the DOPRI5 solver implemented with the adjoint method.



(a) (b) (c) (d)

Fig. 6. Evolution of the OA reached by ResNet (blue), the proposed ODEnet with the DOPRI5 solver and Runge–Kutta integration (orange), and the proposed ODEnet with the DOPRI5 solver and adjoint method (green), considering different amounts of training data. We report the results obtained for (a) IP, (b) UP, (c) SV, and (d) KSC scenes.



(a) (b) (c) (d)

Fig. 7. Evolution of the OA reached by ResNet (blue), the proposed ODEnet with the DOPRI5 solver and Runge–Kutta integration (orange), and the proposed ODEnet with the DOPRI5 solver and adjoint method (green) at different epochs. We report the results obtained for (a) IP, (b) UP, (c) SV, and (d) KSC scenes.

adjoint method better for KSC and SV scenes with low training percentages.

The aforementioned results clearly illustrate the impact that the overfitting of learnable parameters has on the ResNet model, which needs more training data to achieve the same performance as our ODEnet models. Moreover, Fig. 7 shows that this overfitting problem happens at early epochs of the classifiers. Specifically, it can be observed in this figure how the OA obtained by ODEnet increases faster than that achieved by ResNet in the earliest epochs, in particular when complex scenes (such as IP and KSC) are classified.

These observed benefits confirm the following introspections: the ability of the DOPRI5 solver to adapt the model's learning to the complexity of the problem and the significant reduction that can be achieved in terms of the required number of parameters. The latter important benefit is quantitatively measured in Table II, where the number of required model parameters is displayed for each HSI data set. Specifically, the proposed ODEnet and ODEnetAdj models are able to overcome the performance of the traditional ResNet model by

TABLE II

NUMBER OF TRAINABLE PARAMETERS FOR THE STANDARD RESNET MODEL AND THE PROPOSED METHOD IMPLEMENTED WITH DOPRI5 SOLVER AND RUNGE–KUTTA INTEGRATION (ODENET) AND WITH THE ADJOINT METHOD (ODENETADJ)

Dataset	ResNet	ODEnet	ODEnetAdj	Reduction
IP	638416	268624	268624	2.38
UP	582089	212297	212297	2.74
SV	640720	270928	270928	2.36
KSC	624397	254605	254605	2.45

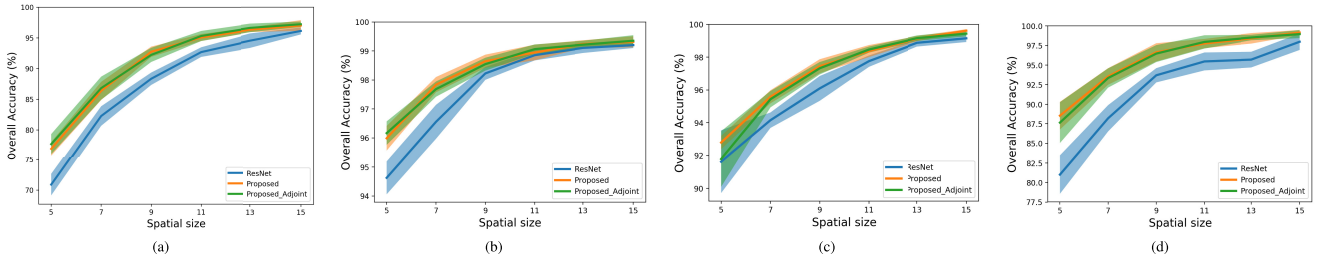


Fig. 8. Evolution of the OA reached by ResNet (blue), the proposed ODEnet the with DOPRI5 solver and Runge–Kutta integration (orange), and the proposed ODEnet with the DOPRI5 solver and adjoint method (green), considering different spatial window sizes. We report the results obtained for (a) IP, (b) UP, (c) SV, and (d) KSC scenes.

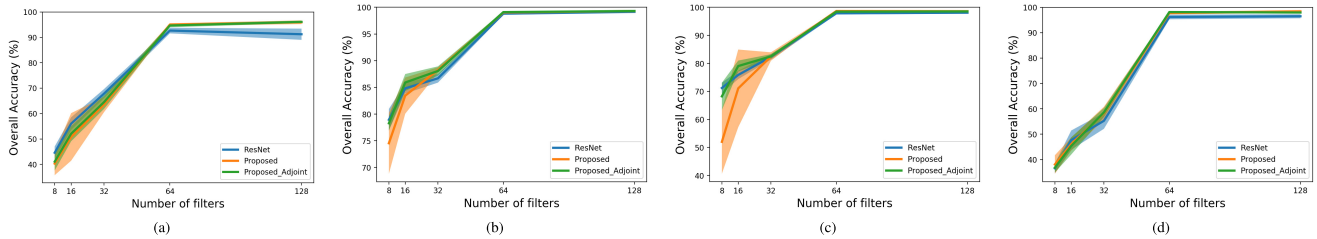


Fig. 9. Evolution of the OA reached by ResNet (blue), the proposed ODEnet with the DOPRI5 solver and Runge–Kutta integration (orange), and the proposed ODEnet with the DOPRI5 solver and adjoint method (green), considering different numbers of filters in each block. We report the results obtained for (a) IP, (b) UP, (c) SV, and (d) KSC scenes.

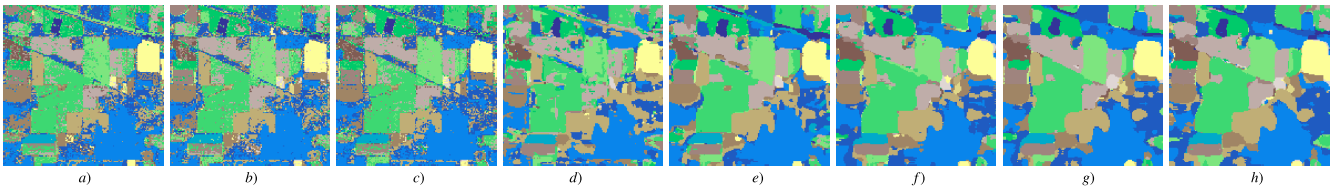


Fig. 10. Classification maps obtained for the IP scene by different classifiers (see Table III). Note that the overall classification accuracies are shown in brackets and the best result is highlighted in bold font. (a) MLR (78.19%). (b) SVM (83.63%). (c) MLP (84.03%). (d) CNN2D (87.16%). (e) CNN3D (95.45%). (f) ResNet (96.55%). (g) ODEnet (97.61%). (h) ODEnetAdj (97.55%).

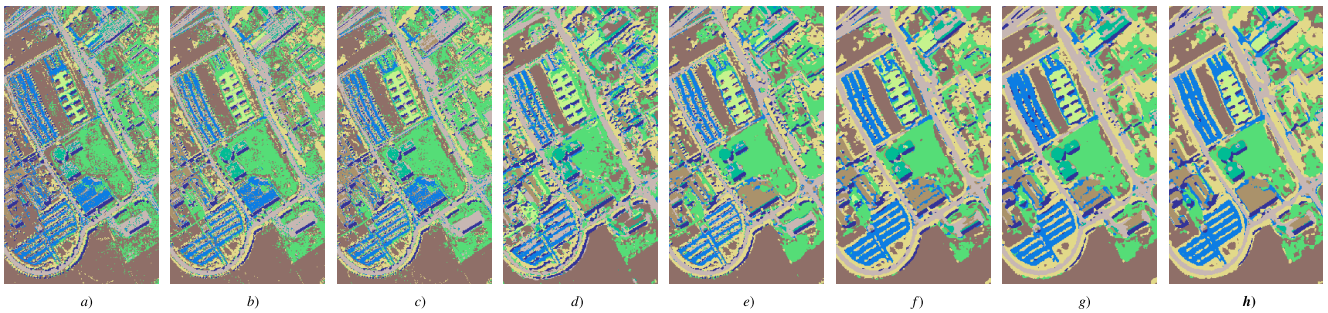


Fig. 11. Classification maps obtained for the UP scene by different classifiers (see Table IV). Note that the overall classification accuracies are shown in brackets and the best result is highlighted in bold font. (a) MLR (89.89%). (b) SVM (94.40%). (c) MLP (94.39%). (d) CNN2D (96.02%). (e) CNN3D (99.02%). (f) ResNet (99.54%). (g) ODEnet (99.67%). (h) ODEnetAdj (99.69%).

using less than half of its training parameters, avoiding quite effectively the overfitting problem.

G. Experiment 4: Testing Different Network Configurations

In this experiment, we report the results obtained by the proposed ODEnet considering different configurations of the model, in particular, the initial amount of information employed by the ODEnet, ODEnetAdj, and ResNet by testing different spatial sizes of the models' input data patch, and the number of features extracted and processed by the convolutional layers.

On the one hand, Fig. 8 shows the obtained results in terms of OA considering input patches comprised of 5×5, 7×7, 9×9, 11×11, 13×13, and 15×15 pixels. As we can observe, the proposed models exhibit very similar behaviors, being able to outperform the accuracy reached by the ResNet in every

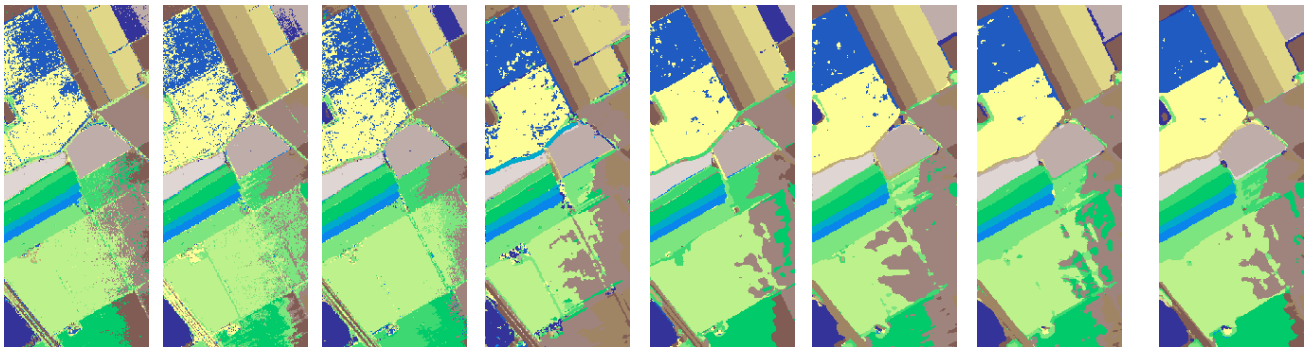


Fig. 12. Classification maps obtained for the SV scene by different classifiers (see Table V). Note that the overall classification accuracies are shown in brackets and the best result is highlighted in bold font. (a) MLR (92.37%). (b) SVM (93.65%). (c) MLP (93.15%). (d) CNN2D (95.27%). (e) CNN3D (98.45%). (f) ResNet (99.28%). (g) ODEnet (99.42%). (h) ODEnetAdj (99.41%).

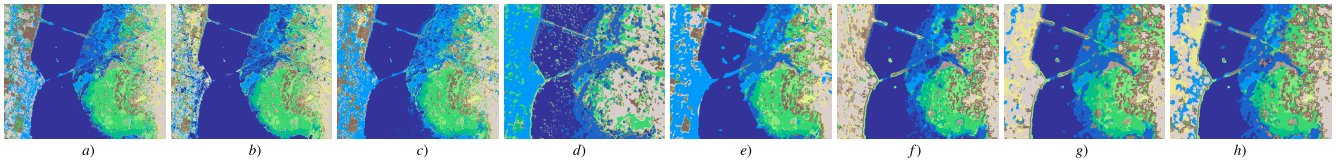


Fig. 13. Classification maps obtained for the KSC scene by different classifiers (see Table VI). Note that the overall classification accuracies are shown in brackets and the best result is highlighted in bold font. (a) MLR (92.74%). (b) SVM (92.92%). (c) MLP (90.22%). (d) CNN2D (66.04%). (e) CNN3D (98.10%). (f) ResNet (98.39%). (g) ODEnet (99.24%). (h) ODEnetAdj (99.03%).

TABLE III

CLASSIFICATION RESULTS OBTAINED BY DIFFERENT METHODS FOR THE IP DATA SET, USING 15% OF THE AVAILABLE LABELED DATA FOR TRAINING AND 11 × 11 INPUT SPATIAL PATCH SIZE

Class	MLR	SVM	MLP	CNN2D	CNN3D	ResNet	ODEnet	ODEnetAdj
0	74.59±0.48	81.36±0.42	81.23±0.96	85.61±1.09	95.34±0.72	96.07±0.55	**97.28**±0.34	97.21±0.38
1	33.33±10.51	56.92±12.96	65.38±13.96	75.38±17.7	80.77±15.44	86.41±9.87	93.08±5.13	**93.59**±5.29
2	76.36±1.9	81.12±1.09	78.92±2.09	84.63±2.56	94.68±1.84	94.15±1.02	96.02±1.43	**96.08**±0.97
3	57.3±2.15	74.03±2.01	68.14±3.81	77.63±3.52	95.32±1.44	94.4±2.79	**97.15**±1.44	97.09±1.69
4	43.03±6.83	61.29±4.81	73.78±4.65	84.13±5.09	91.84±3.68	**96.77**±2.48	96.62±2.61	96.32±2.72
5	86.73±3.94	89.71±3.83	88.46±2.8	83.29±5.47	96.15±2.07	97.24±1.21	96.63±1.7	**97.63**±2.86
6	96.23±0.9	97.0±1.13	95.13±1.66	90.47±2.48	99.11±0.47	97.71±1.08	**99.18**±0.38	98.76±0.59
7	54.78±7.33	77.39±10.07	82.61±7.78	84.78±16.53	80.87±17.95	80.87±12.17	96.52±5.43	**96.96**±4.37
8	98.45±0.96	97.73±1.87	98.69±1.13	97.17±2.27	99.73±0.58	99.43±1.09	**99.9**±0.16	99.9±0.16
9	18.24±13.27	50.59±11.22	64.71±13.67	**85.88**±11.53	83.53±17.41	67.65±18.27	82.94±10.0	77.65±9.41
10	65.3±2.2	76.42±1.97	78.98±3.67	78.75±3.78	94.49±2.55	95.33±1.79	**97.53**±1.35	96.38±1.46
11	79.87±1.71	84.21±1.65	83.25±2.19	90.48±1.53	96.87±1.19	97.77±0.72	98.23±0.47	**98.26**±0.43
12	61.23±2.31	77.64±2.14	79.58±4.16	76.25±3.54	89.48±4.18	94.01±1.81	94.11±2.46	**95.32**±1.49
13	98.51±0.64	98.33±1.13	98.1±1.06	98.33±1.13	**99.89**±0.23	99.54±0.5	99.25±0.85	99.6±0.45
14	94.99±1.31	94.51±1.71	95.79±0.92	97.4±0.95	98.91±0.47	99.09±0.87	99.23±1.07	**99.29**±0.7
15	63.2±3.62	62.93±4.7	65.88±3.48	89.7±3.71	91.68±4.25	97.13±2.58	**97.65**±1.45	96.55±2.62
16	86.08±2.59	87.47±4.22	94.05±4.57	93.42±5.39	**98.48**±2.18	93.92±5.18	96.84±3.93	95.7±3.81
OA (%)	77.87±0.42	83.68±0.38	83.57±0.85	87.43±0.95	95.92±0.63	96.55±0.48	**97.61**±0.3	97.55±0.33
AA (%)	69.6±0.59	79.21±1.49	81.97±1.76	86.73±1.91	93.24±1.61	93.21±1.45	**96.3**±0.87	95.94±1.09
Kappa (x100)	74.59±0.48	81.36±0.42	81.23±0.96	85.61±1.09	95.34±0.72	96.07±0.55	**97.28**±0.34	97.21±0.38
Runtime (s)	5.99±1.03	**0.28**±0.01	99.49±5.46	66.55±4.16	172.89±15.73	81.63±0.13	192.6±0.19	257.55±2.62

scene, in particular when the spatial windows are very small. Moreover, the improvement in the OA's values increases as the spatial windows' size increases. However, while the difference, in terms of accuracy, between small spatial windows is very pronounced (for instance, between windows of 5 × 5 and 9 × 9 pixels, there are approximately ten percentage points of improvement in IP and KSC and four percentage points in UP and SV), between bigger windows, the difference is noticeably smaller (for instance, between windows of 11 × 11 and 15 × 15). In this sense, as the amount of information to be processed increases with the dimensions of the input data patch, increasing also both memory requirements and computation times, we consider paths of 11 × 11 pixels as an optimal input data size, with a good ratio between performance and computing time.

On the other hand, Fig. 9 shows the obtained results in terms of OA too, considering input patches of 11 × 11 and convolutional layers with 8, 16, 32, 64, and 128 filters. As we can observe for each data set, the OA increases its value as more filters are added. In particular, the best

TABLE IV

CLASSIFICATION RESULTS OBTAINED BY DIFFERENT METHODS FOR THE UP DATA SET, USING 10% OF THE AVAILABLE LABELED DATA FOR TRAINING AND 11 × 11 INPUT SPATIAL PATCH SIZE

Class	MLR	SVM	MLP	CNN2D	CNN3D	ResNet	ODEnet	ODEnetAdj
0	86.42±0.2	92.56±0.14	92.42±0.25	94.72±0.29	99.08±0.13	99.39±0.11	99.56±0.1	**99.59**±0.11
1	92.31±0.62	94.46±0.58	94.98±0.72	95.12±0.99	99.19±0.38	99.44±0.25	**99.64**±0.17	99.59±0.3
2	96.07±0.36	98.31±0.18	97.82±0.3	98.21±0.33	99.89±0.06	99.87±0.09	**99.93**±0.06	99.89±0.05
3	74.0±1.86	79.37±1.84	80.32±2.92	90.33±1.45	96.88±0.79	98.3±1.18	98.89±0.52	**99.08**±0.65
4	88.49±0.96	94.55±0.58	93.59±1.65	97.6±0.54	99.18±0.39	**99.23**±0.38	99.22±0.38	99.13±0.44
5	99.31±0.27	99.28±0.18	99.54±0.15	99.16±0.49	**100.0**±0.0	99.98±0.05	99.98±0.05	99.98±0.05
6	77.42±0.84	88.86±1.06	89.93±1.62	92.01±0.96	**99.88**±0.14	99.62±0.37	99.82±0.18	99.87±0.22
7	57.03±3.04	85.41±2.02	85.98±1.94	83.54±3.16	93.85±2.17	98.08±0.74	98.48±1.2	**98.77**±1.04
8	86.77±0.76	90.67±1.11	89.43±1.83	96.63±0.77	98.84±0.24	99.37±0.38	99.4±0.36	**99.65**±0.24
9	99.77±0.1	**99.91**±0.1	99.89±0.1	98.92±0.87	99.82±0.18	98.98±1.1	99.33±0.75	99.72±0.24
OA (%)	89.84±0.16	94.41±0.1	94.3±0.19	96.02±0.22	99.31±0.1	99.54±0.08	99.67±0.08	**99.69**±0.08
AA (%)	85.69±0.33	92.31±0.24	92.39±0.37	94.61±0.47	98.62±0.26	99.21±0.19	99.41±0.22	**99.52**±0.1
Kappa (x100)	86.42±0.2	92.56±0.14	92.42±0.25	94.72±0.29	99.08±0.13	99.39±0.11	99.56±0.1	**99.59**±0.11
Runtime (s)	9.11±1.57	0.4±0.01	439.55±22.77	306.69±19.04	409.88±65.98	160.17±0.43	508.68±1.71	671.88±3.54

TABLE V

CLASSIFICATION RESULTS OBTAINED BY DIFFERENT METHODS FOR THE SV DATA SET, USING 10% OF THE AVAILABLE LABELED DATA FOR TRAINING AND 11 × 11 INPUT SPATIAL PATCH SIZE

Class	MLR	SVM	MLP	CNN2D	CNN3D	ResNet	ODEnet	ODEnetAdj
0	91.55±0.11	92.94±0.16	92.37±0.17	94.72±0.49	98.23±0.21	99.2±0.18	**99.35**±0.16	99.35±0.19
1	99.49±0.24	99.57±0.28	99.52±0.37	97.2±2.3	**100.0**±0.0	99.84±0.24	99.91±0.1	99.91±0.1
2	99.92±0.06	99.79±0.13	99.89±0.11	98.96±1.04	**99.98**±0.04	99.91±0.13	99.86±0.21	99.88±0.19
3	99.45±0.25	99.58±0.16	99.15±0.51	96.66±3.15	**99.94**±0.07	99.85±0.14	99.91±0.17	99.8±0.25
4	99.25±0.21	99.3±0.38	99.31±0.34	99.79±0.39	99.67±0.28	99.44±0.62	99.68±0.43	**99.83**±0.17
5	99.07±0.23	98.69±0.47	99.13±0.26	98.75±1.29	99.24±0.38	**99.86**±0.16	99.83±0.25	99.85±0.16
6	99.94±0.05	99.85±0.09	99.86±0.14	99.35±0.46	100.0±0.01	**100.0**±0.0	100.0±0.0	100.0±0.01
7	99.68±0.17	99.71±0.16	99.62±0.25	99.51±0.36	99.74±0.28	99.88±0.11	**99.97**±0.05	99.92±0.11
8	87.57±0.6	88.38±0.63	87.1±1.06	92.47±1.43	96.54±0.5	98.42±0.39	**98.82**±0.38	98.62±0.41
9	99.8±0.15	99.72±0.13	99.87±0.17	99.13±0.44	99.96±0.08	**99.98**±0.02	99.96±0.07	99.96±0.04
10	95.41±0.88	96.16±0.71	96.34±0.81	96.66±1.03	99.19±0.34	99.59±0.25	**99.75**±0.32	99.48±0.64
11	97.54±0.91	97.8±0.81	98.3±1.65	97.64±1.14	99.56±0.23	**99.84**±0.31	99.69±0.51	99.4±0.48
12	99.68±0.2	99.74±0.21	99.73±0.16	97.76±1.41	**100.0**±0.0	99.97±0.07	99.99±0.02	99.99±0.02
13	99.14±0.4	98.68±0.68	98.39±1.37	98.68±0.77	99.65±0.79	99.58±0.6	99.62±0.54	**99.82**±0.29
14	96.87±1.07	96.23±1.87	96.81±1.41	97.65±1.18	99.17±0.64	99.69±0.41	**99.89**±0.28	99.85±0.19
15	67.47±0.57	75.4±1.08	73.2±2.13	85.05±2.24	94.79±1.13	97.91±0.67	98.08±0.47	**98.49**±0.73
16	98.64±0.42	98.76±0.37	98.79±0.43	93.37±1.94	99.51±0.35	99.58±0.45	**99.79**±0.18	99.76±0.19
OA (%)	92.42±0.1	93.66±0.14	93.15±0.15	95.26±0.44	98.41±0.19	99.28±0.16	**99.42**±0.14	99.41±0.18
AA (%)	96.18±0.08	96.71±0.09	96.56±0.18	96.79±0.24	99.18±0.1	99.58±0.12	**99.67**±0.09	99.66±0.09
Kappa (x100)	91.55±0.11	92.94±0.16	92.37±0.17	94.72±0.49	98.23±0.21	99.2±0.18	**99.35**±0.16	99.35±0.19
Runtime (s)	50.49±0.22	1.3±0.02	689.03±29.61	457.91±12.95	826.38±58.19	239.64±0.18	730.73±36.55	972.9±36.21

OA is reached with 64 filters, remaining quite similar with 128 filters. Actually, the OA is improved very slightly with 128 filters; however, the computational cost of this network's configuration is considerably higher than with 64 filters. For this reason, we consider 64 to be the optimum number of filters for each convolutional layer.

H. Experiment 5: Testing Different HSI Classifiers

Our final experiment compares our proposed ODEnet models with some widely used classifiers available in the HSI classification literature. Fig. 10 (IP), Fig. 11 (UP), Fig. 12 (SV), and Fig. 13 (KSC) show the classification maps obtained by each considered method, while Table III (IP), Table IV (UP), Table V (SV), and Table VI (KSC) give the individual class accuracies and the global OA, AA, and Kappa values obtained by each classifier with the corresponding standard deviations, respectively, including also the obtained runtimes of each experiment.

As a general comment, the improvement introduced by spatial and spectral–spatial models over pixelwise classifiers is remarkable. For instance, CNN2D introduces around 2% points of improvement in OA when compared to the most accurate spectral model, i.e., the SVM (for UP and SV) and the MLP (for IP), with an exception in the KSC scene, in which the spatial information appears to be not enough discriminatory to carry out an accurate classification, as we can observe in Table VI and the corresponding classification maps in Fig. 13. The limitations of pixelwise and spatial-based classifiers can be easily overcome by spectral–spatial classifiers, where the combination of spectral and spatial–contextual information is able to significantly reduce the uncertainty and data variability of HSI pixels, as it can be observed on complex data sets, such as IP (see Table III) and, particularly, KSC (see Table VI). This results in better classification maps, where the "salt and pepper" classification noise is practically removed. However, it is interesting to focus on the classification maps produced by the spatial–spectral CNN3D classifier

TABLE VI
CLASSIFICATION RESULTS OBTAINED BY DIFFERENT METHODS FOR THE KSC DATA SET, USING 15% OF THE AVAILABLE LABELED DATA FOR TRAINING AND 11 × 11 INPUT SPATIAL PATCH SIZE

Class	MLR	SVM	MLP	CNN2D	CNN3D	ResNet	ODEnet	ODEnetAdj
0	91.57±0.76	92.52±0.57	88.87±0.55	61.8±1.35	98.22±0.27	98.2±0.59	**99.16**±0.24	98.93±0.42
1	95.0±0.96	95.57±1.31	96.53±0.98	96.35±1.93	**100.0**±0.0	99.47±0.82	100.0±0.0	99.83±0.26
2	93.01±2.24	91.55±3.06	84.76±2.08	37.23±6.03	95.78±2.86	96.7±2.74	98.93±1.55	**99.22**±0.66
3	88.99±1.73	88.89±3.52	88.06±3.62	21.15±8.09	97.37±1.6	96.31±3.35	**99.12**±1.33	96.59±3.74
4	71.17±5.63	74.91±4.73	60.84±5.86	21.26±7.13	86.4±3.18	88.79±2.1	**95.09**±1.91	94.49±4.34
5	71.1±6.49	73.75±4.89	59.56±5.35	57.87±5.07	90.0±4.45	91.99±8.4	90.51±5.11	**92.06**±5.44
6	70.77±7.17	74.79±4.34	58.09±2.73	40.26±14.57	96.6±2.14	98.2±1.09	**98.92**±1.41	98.81±1.56
7	81.57±5.8	86.18±5.1	84.04±4.71	30.79±29.78	99.21±1.33	97.87±2.22	**99.89**±0.34	99.33±1.68
8	91.72±1.84	92.76±2.49	88.5±0.87	35.41±4.06	98.93±1.22	98.91±1.05	99.73±0.35	**99.86**±0.18
9	96.54±1.18	96.92±1.54	96.04±1.43	71.0±4.87	**99.91**±0.21	99.32±0.93	99.59±0.38	99.68±0.61
10	96.27±1.52	97.32±1.91	93.62±1.91	52.94±5.82	99.85±0.44	99.62±0.39	**99.91**±0.19	99.56±1.04
11	97.58±0.92	96.99±2.22	96.99±1.24	95.28±2.07	**99.97**±0.08	99.89±0.34	99.92±0.25	99.55±0.69
12	94.54±3.19	96.21±1.31	92.22±1.16	61.45±6.42	99.65±0.42	99.02±1.17	**99.93**±0.15	99.51±0.74
13	**100.0**±0.0	100.0±0.0	99.72±0.34	91.39±3.37	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0
OA (%)	92.43±0.68	93.29±0.51	90.02±0.5	66.03±1.23	98.4±0.24	98.39±0.53	**99.24**±0.21	99.03±0.37
AA (%)	88.33±1.0	89.68±0.87	84.54±0.87	54.8±2.42	97.21±0.48	97.39±0.93	**98.58**±0.49	98.35±0.71
Kappa (x100)	91.57±0.76	92.52±0.57	88.87±0.55	61.8±1.35	98.22±0.27	98.2±0.59	**99.16**±0.24	98.93±0.42
Runtime (s)	2.89±0.24	0.05±0.01	87.98±3.74	66.91±4.63	73.17±1.36	57.47±0.08	114.57±0.17	154.55±7.2

(for instance in Figs. 11 and 13), where multiple patches have been wrongly labeled, obtaining visually noisy classification maps. This can be observed in the lower leftmost corner of the KSC scene (see Fig. 13), where the vast majority of pixels have been missclassified as Salt-marsh. These deficiencies are highlighted by the differences between the OA and AA values, where the AA is several percentual points lower, indicating the existence of an overfitting problem (see Tables III and VI).

Adding residual learning via ResNet can improve the accuracy results, reducing the gap between the OA and AA on some HSI data sets, such as UP (see Table IV), SV (see Table V), and KSC (see Table VI), and improving the visual appearance of the corresponding classification maps. However, this gap between the OA and AA scores cannot be reduced by ResNet in the IP scene (in fact, for this scene, the gap becomes larger). In turn, the proposed ODEnet models are able to reach very similar OA, AA, and Kappa values in all the cases, exhibiting very good consistency in terms of model performance with higher robustness on the obtained results. As we can observe, the proposed method is able to reach the best accuracy scores in all the considered data sets, visually clean classification maps, where the number of missclassified patches is drastically reduced.

If we now focus on the execution times reported in Table III (IP), Table IV (UP), Table V (SV), and Table VI (KSC), it can be observed that pixelwise methods are faster than spatial and spectral–spatial ones, being SVM the fastest classifier. The computational cost of the proposed ODEnet model is higher when compared to the other deep models (CNN2D, CNN3D, and ResNet), mainly due to the great optimization performed by the frameworks in which these classifiers have been implemented. In this regard, it is necessary to conduct an effort to optimize the code of the ODEsolver in order to provide a more efficient version, although (as shown in our second experiment) the use of higher tolerance values allows for a significant reduction of computation times.

IV. CONCLUSION AND FUTURE WORK

This article proposes, for the first time in the literature, a redefinition of the traditional discrete-layer ResNet model as a continuous-time evolving model through the implementation of an ODE parameterized by a neural network with the aim of improving the classification of remotely sensed HSI data by producing better and more robust feature representations.

The obtained experimental results, conducted using four widely used HSI data sets, demonstrate the significant benefits and improvements introduced by the proposed method, which are able to reach consistently higher accuracy values in comparison with the traditional ResNet model, at the same time it significantly reduces the number of parameters that need to be used and fine-tuned, providing a highly efficient mechanism to address the problems of overfitting and data degradation in very deep networks. Moreover, the integration of adaptive solvers, such as DOPRI5, offers great flexibility when processing and classifying complex HSI scenes, allowing the model to obtain highly refined features for classification purposes.

Encouraged by the good results obtained in terms of model's accuracy, in the future, we will develop an optimized and parallelized implementation of the proposed ODEnet, exploring other solver algorithms in order to reduce the computational complexity.

ACKNOWLEDGMENT

The authors would like to thank the Associate Editor and the three anonymous reviewers for their outstanding comments and suggestions, which greatly helped us to improve the technical quality and presentation of this article.

REFERENCES

[1] J. A. Richards, *Remote Sensing Digital Image Analysis*, vol. 3. Berlin, Germany: Springer, 1999.

[2] X.-L. Chen, H.-M. Zhao, P.-X. Li, and Z.-Y. Yin, "Remote sensing image-based analysis of the relationship between urban heat island and land use/cover changes," *Remote Sens. Environ.*, vol. 104, no. 2, pp. 133–146, 2006.

[3] D. A. Mariano *et al.*, "Use of remote sensing indicators to assess effects of drought and human-induced land degradation on ecosystem health in Northeastern Brazil," *Remote Sens. Environ.*, vol. 213, pp. 129–143, Aug. 2018.

[4] T. Lillesand, R. W. Kiefer, and J. Chipman, *Remote Sensing and Image Interpretation*. Hoboken, NJ, USA: Wiley, 2014.

[5] R. O. Green *et al.*, "Imaging spectroscopy and the airborne visible/infrared imaging spectrometer (AVIRIS)," *Remote Sens. Environ.*, vol. 65, no. 3, pp. 227–248, Sep. 1998.

[6] D. J. Mulla, "Twenty five years of remote sensing in precision agriculture: Key advances and remaining knowledge gaps," *Biosyst. Eng.*, vol. 114, no. 4, pp. 358–371, Apr. 2013.

[7] M. Govender, K. Chetty, and H. Bulcock, "A review of hyperspectral remote sensing and its application in vegetation and water resource studies," *Water SA*, vol. 33, no. 2, pp. 145–151, 2007.

[8] B. Hörig, F. Kühn, F. Oschütz, and F. Lehmann, "HyMap hyperspectral remote sensing to detect hydrocarbons," *Int. J. Remote Sens.*, vol. 22, no. 8, pp. 1413–1422, 2001.

[9] P. M. Treitz and P. J. Howarth, "Hyperspectral remote sensing for estimating biophysical parameters of forest ecosystems," *Progr. Phys. Geography*, vol. 23, no. 3, pp. 359–390, 1999.

[10] V. Sander *et al.*, "Hyperspectral remote sensing of fire: State-of-the-art and future perspectives," *Remote Sens. Environ.*, vol. 216, pp. 105–121, Oct. 2018.

[11] M. Herold, D. A. Roberts, M. E. Gardner, and P. E. Dennison, "Spectrometry for urban area remote sensing—Development and analysis of a spectral library from 350 to 2400 nm," *Remote Sens. Environ.*, vol. 91, nos. 3–4, pp. 304–319, 2004.

[12] P. W. Yuen and M. Richardson, "An introduction to hyperspectral imaging and its application for security, surveillance and target acquisition," *Imag. Sci. J.*, vol. 58, no. 5, pp. 241–253, Oct. 2010.

[13] J. Haut, M. Paoletti, J. Plaza, and A. Plaza, "Cloud implementation of the K-means algorithm for hyperspectral image analysis," *J. Supercomput.*, vol. 73, no. 1, pp. 514–529, 2017.

[14] 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.

[15] J. Li, J. M. Bioucas-Dias, and A. Plaza, "Semisupervised hyperspectral image classification using soft sparse multinomial logistic regression," *IEEE Geosci. Remote Sens. Lett.*, vol. 10, no. 2, pp. 318–322, Mar. 2013.

[16] P. Ghamisi *et al.*, "Advances in hyperspectral image and signal processing: A comprehensive overview of the state of the art," *IEEE Geosci. Remote Sens. Mag.*, vol. 5, no. 4, pp. 37–78, Dec. 2017.

[17] P. Ghamisi, J. Plaza, Y. Chen, J. Li, and A. J. Plaza, "Advanced spectral classifiers for hyperspectral images: A review," *IEEE Geosci. Remote Sens. Mag.*, vol. 5, no. 1, pp. 8–32, Mar. 2017.

[18] P. K. Goel, S. O. Prasher, R. M. Patel, J.-A. Landry, R. Bonnell, and A. A. Viau, "Classification of hyperspectral data by decision trees and artificial neural networks to identify weed stress and nitrogen status of corn," *Comput. Electron. Agricult.*, vol. 39, no. 2, pp. 67–93, 2003.

[19] F. Ratle, G. Camps-Valls, and J. Weston, "Semisupervised neural networks for efficient hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 5, pp. 2271–2282, May 2010.

[20] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Math. Control, Signals Syst.*, vol. 2, no. 4, pp. 303–314, 1989.

[21] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Netw.*, vol. 4, no. 2, pp. 251–257, 1991.

[22] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[23] 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, Mar. 2018.

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

[25] S. Kaewpijit, J. L. Moigne, and T. El-Ghazawi, "Automatic reduction of hyperspectral imagery using wavelet spectral analysis," *IEEE Trans. Geosci. Remote Sens.*, vol. 41, no. 4, pp. 863–871, Apr. 2003.

[26] J. M. Haut, M. E. Paoletti, J. Plaza, and A. Plaza, "Fast dimensionality reduction and classification of hyperspectral images with extreme learning machines," *J. Real-Time Image Process.*, vol. 15, no. 3, pp. 439–462, 2018.

[27] D. J. Field, "Wavelets, vision and the statistics of natural scenes," *Phil. Trans. Roy. Soc. London A, Math., Phys. Eng. Sci.*, vol. 357, no. 1760, pp. 2527–2542, 1999.

[28] M. E. Paoletti *et al.*, "Capsule networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 4, pp. 2145–2160, Apr. 2019.

[29] W. Hu, Y. Huang, L. Wei, F. Zhang, and H. Li, "Deep convolutional neural networks for hyperspectral image classification," *J. Sensors*, vol. 2015, Jan. 2015, Art. no. 258619.

[30] J. Yue, W. Zhao, S. Mao, and H. Liu, "Spectral–spatial classification of hyperspectral images using deep convolutional neural networks," *Remote Sens. Lett.*, vol. 6, no. 6, pp. 468–477, Jun. 2015.

[31] 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.

[32] 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.

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

[34] M. Lin, Q. Chen, and S. Yan, "Network in network," 2013, *arXiv:1312.4400*. [Online]. Available: https://arxiv.org/abs/1312.4400

[35] N. He *et al.*, "Feature extraction with multiscale covariance maps for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 2, pp. 755–769, Feb. 2019.

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

[37] 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.

[38] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[39] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.

[40] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2377–2385.

[41] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," 2015, *arXiv:1505.00387*. [Online]. Available: https://arxiv.org/abs/1505.00387

[42] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 4700–4708.

[43] K. Zhang, M. Sun, T. X. Han, X. Yuan, L. Guo, and T. Liu, "Residual networks of residual networks: Multilevel residual networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 6, pp. 1303–1314, Jun. 2018.

[44] J. M. Haut, R. Fernandez-Beltran, M. E. Paoletti, J. Plaza, A. Plaza, and F. Pla, "A new deep generative network for unsupervised remote sensing single-image super-resolution," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 11, pp. 6792–6810, Nov. 2018.

[45] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 630–645.

[46] 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, Feb. 2018.

[47] 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, Feb. 2019.

[48] M. E. Paoletti, J. M. Haut, J. Plaza, and A. Plaza, "Deep&dense convolutional neural network for hyperspectral image classification," *Remote Sens.*, vol. 10, no. 9, p. 1454, 2018.

[49] M. Thorpe and Y. van Gennip, "Deep limits of residual neural networks," 2018, *arXiv:1810.11741*. [Online]. Available: https://arxiv.org/abs/1810.11741

[50] L. Ruthotto and E. Haber, "Deep neural networks motivated by partial differential equations," 2018, *arXiv:1804.04272*. [Online]. Available: https://arxiv.org/abs/1804.04272

[51] E. Weinan, "A proposal on machine learning via dynamical systems," *Commun. Math. Statist.*, vol. 5, no. 1, pp. 1–11, 2017.

[52] E. Haber, L. Ruthotto, E. Holtham, and S.-H. Jun, "Learning across scales—Multiscale methods for convolution neural networks," in *Proc. 32nd AAAI Conf. Artif. Intell.*, Apr. 2018, pp. 3142–3148.
[53] Y. Lu, A. Zhong, Q. Li, and B. Dong, "Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations," 2017, *arXiv:1710.10121*. [Online]. Available: https://arxiv.org/abs/1710.10121
[54] T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 6572–6583.
[55] M. E. Paoletti, J. M. Haut, J. Plaza, and A. Plaza, "An investigation on self-normalized deep neural networks for hyperspectral image classification," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2018, pp. 3607–3610.
[56] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, vol. 1. Cambridge, MA, USA: MIT Press, 2016.
[57] 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.
[58] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *SIAM Rev.*, vol. 60, no. 2, pp. 223–311, 2018.
[59] J. D. Lambert, *Numerical Methods for Ordinary Differential Systems: The Initial Value Problem*. Hoboken, NJ, USA: Wiley, 1991.
[60] J. R. Dormand and P. J. Prince, "A family of embedded Runge-Kutta formulae," *J. Comput. Appl. Math.*, vol. 6, no. 1, pp. 19–26, 1980.
[61] K. Atkinson, *An Introduction to Numerical Analysis*. Hoboken, NJ, USA: Wiley, 2008.
[62] J. T. Betts, "Survey of numerical methods for trajectory optimization," *J. Guid. Control Dyn.*, vol. 21, no. 2, pp. 193–207, Mar./Apr. 1998.
[63] 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," *Proc. SPIE*, vol. 0868, pp. 134–141, Apr. 1988.

Mercedes E. Paoletti (S'17) received the B.Sc. and M.Sc. degrees in computer engineering from the University of Extremadura, Cáceres, Spain, in 2014 and 2016, respectively, where she is currently pursuing the Ph.D. degree with the University Teacher Training Programme from the Spanish Ministry of Education.

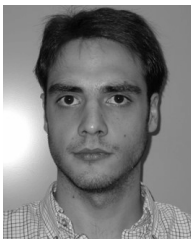
She is currently a member of the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, University of Extremadura. Her research interests include remote sensing and analysis of very high spectral resolution with the current focus on deep learning and high performance computing.

Ms. Paoletti was a recipient of the 2019 Outstanding Paper Award Recognition in the WHISPERS 2019 Congress. She has been a Reviewer of the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING and the IEEE GEOSCIENCE AND REMOTE SENSING LETTERS.

Juan Mario Haut (S'17–M'19) received the B.Sc. and M.Sc. degrees in computer engineering from the University of Extremadura, Cáceres, Spain, in 2011 and 2014, respectively, and the Ph.D. degree in information technology with the University Teacher Training Programme from the Spanish Ministry of Education, University of Extremadura in 2019.

He is a member of the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, University of Extremadura. His research interests include remote sensing and analysis of very high spectral resolution with the current focus on machine (deep) learning and cloud computing.

Dr. Haut was a recipient of the Best Reviewers Recognition Award of the IEEE GEOSCIENCE AND REMOTE SENSING LETTERS in 2019 and the Outstanding Paper Award in the Whispers 2019 Congress. He has been a Reviewer of the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING, and the IEEE GEOSCIENCE AND REMOTE SENSING LETTERS.



Javier Plaza (M'09–SM'15) received the M.Sc. and Ph.D. degrees in computer engineering from the University of Extremadura, Cáceres, Spain, in 2004 and 2008, respectively.

He is a member of the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, University of Extremadura. He has authored more than 150 publications, including over 50 JCR journal articles, ten book chapters, and 90 peer-reviewed conference proceeding articles. His main research interests include hyperspectral data processing and parallel computing of remote sensing data.

Dr. Plaza was a recipient of the Outstanding Ph.D. Dissertation Award at the University of Extremadura in 2008, the Best Column Award of the *IEEE Signal Processing Magazine* in 2015, the most highly cited paper (2005–2010) in the *Journal of Parallel and Distributed Computing*, and the best paper awards at the IEEE International Conference on Space Technology and the IEEE Symposium on Signal Processing and Information Technology. He has guest edited four special issues on hyperspectral remote sensing for different journals. He is an Associate Editor of the IEEE GEOSCIENCE AND REMOTE SENSING LETTERS and the IEEE REMOTE SENSING CODE LIBRARY.

Antonio Plaza (M'05–SM'07–F'15) received the M.Sc. and Ph.D. degrees in computer engineering from the University of Extremadura, Cáceres, Spain, in 1999 and 2002, respectively

He is currently the Head of the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, University of Extremadura. He has authored more than 600 publications, including over 200 JCR journal articles (over 160 in IEEE journals), 23 book chapters, and around 300 peer-reviewed conference proceeding articles. His research interests include hyperspectral data processing and parallel computing of remote sensing data.

Dr. Plaza is a fellow of the IEEE for the contributions to hyperspectral data processing and parallel computing of earth observation data. He was a member of the Editorial Board of the IEEE GEOSCIENCE AND REMOTE SENSING NEWSLETTER from 2011 to 2012 and the *IEEE Geoscience and Remote Sensing Magazine* in 2013. He was a recipient of the recognition of Best Reviewers of the IEEE GEOSCIENCE AND REMOTE SENSING LETTERS in 2009 and the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING in 2010, for which he served as an Associate Editor from 2007 to 2012. He was a recipient of the Best Column Award of the *IEEE Signal Processing Magazine* in 2015, the 2013 Best Paper Award of the JSTARS Journal, the most highly cited article (2005–2010) in the *Journal of Parallel and Distributed Computing*, and the best paper awards at the IEEE International Conference on Space Technology and the IEEE Symposium on Signal Processing and Information Technology. He has guest-edited ten special issues on hyperspectral remote sensing for different journals. He is also an Associate Editor of IEEE ACCESS (receiving a recognition as an outstanding Associate Editor of the journal in 2017). He was also a member of the Steering Committee of the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING (JSTARS). He served as the Director of Education Activities for the IEEE Geoscience and Remote Sensing Society (GRSS) from 2011 to 2012, and as the President of the Spanish Chapter of the IEEE GRSS from 2012 to 2016. He has reviewed more than 500 manuscripts for over 50 different journals. He served as the Editor-in-Chief for the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING from 2013 to 2017.

Escuela Politecnica
Av. de la Universidad, S/N, 10003
Caceres, Spain
Phone: 0034927257000. Ext. 51662
Email: aplaza,jplaza{@unex.es}

Dr. Antonio Plaza Miguel y Dr. Javier Plaza Miguel como directores de la tesis titulada "Procesamiento eficiente y profundo de imgenes hiperespectrales de la observación remota de la Tierra y aplicaciones en tareas de clasificación", certifico el factor de impacto y la categorización de la siguiente publicación, incluida en la tesis doctoral. Del mismo modo, se especifica la aportación del doctorado. Si necesitas cualquier información o clarificación, por favor, no dude en contactar conmigo.

Antonio Plaza Miguel PhD. and Javier Plaza Miguel PhD as directors of the Phd thesis titled "Deep-efficient processing of remote sensing hyperspectral images and applications in classification tasks.", certify the impact factor and the categorization of the following publication, included in the doctoral thesis. In the same way, the contribution of the doctorate is specified. If you need any further information or clarification, please do not hesitate contacting me.

Artculo / Paper

Autores/Authors: **M. E. Paoletti**, J. M. Haut, R. Fernandez-Beltran, J. Plaza, A. Plaza, J. Li and F. Pla.
Title: Capsule Networks for Hyperspectral Image Classification.
Journal: IEEE Transactions on Geoscience and Remote Sensing.
Other Information: vol. 57, no. 4, pp. 2145-2160, April 2019.
DOI: 10.1109/TGRS.2018.2871782.
Impact factor 2018: 5.630. Q1
Abstract: Convolutional neural networks (CNNs) have recently exhibited an excellent performance in hyperspectral image classification tasks. However, the straightforward CNN-based network architecture still finds obstacles when effectively exploiting the relationships between hyperspectral imaging (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 a 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 that redefines 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 data sets 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.

Contribución del doctorado: Planteamiento de la hipótesis, desarrollo práctico, análisis y discusión de los resultados, elaboración y escritura del manuscrito.

Firma / Signature
Mayo / May, 2020

Antonio Plaza Miguel

Javier Plaza Miguel

Capsule Networks for Hyperspectral Image Classification

Mercedes E. Paoletti, *Student Member, IEEE*, Juan Mario Haut, *Student Member, IEEE*, Ruben Fernandez-Beltran, Javier Plaza, *Senior Member, IEEE*, Antonio Plaza, *Fellow, IEEE*, Jun Li, *Senior Member, IEEE*, and Filiberto Pla

***Abstract*— Convolutional neural networks (CNNs) have recently exhibited an excellent performance in hyperspectral image classification tasks. However, the straightforward CNN-based network architecture still finds obstacles when effectively exploiting the relationships between hyperspectral imaging (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 a 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 that redefines 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 data sets 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*— Capsule networks (CapsNets), convolutional neural networks (CNNs), hyperspectral imaging (HSI).**

I. Introduction

THE constant development of spectral imaging acquisition technologies, together with the increasing availability of remote sensing platforms, provides plenty of opportunities to manage the detailed spectral–spatial information of the earth's surface [1]–[3]. As a result, the classification of remotely sensed hyperspectral images 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 hyperspectral imaging (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 [13], random forest (RF) [14], extreme learning machines [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: 1) high data complexity and 2) 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 particu-

Manuscript received June 9, 2018; revised August 22, 2018; accepted September 19, 2018. Date of publication October 25, 2018; date of current version March 25, 2019. This work was supported in part by the 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, and en el marco del Plan Estatal de Investigación Científica y Técnica y de Innovación 2013–2016), in part by the Junta de Extremadura (decreto 297/2014, ayudas para la realización de actividades de investigación y desarrollo tecnológico, and de divulgación y de transferencia de conocimiento por los Grupos de Investigación de Extremadura) under Grant GR15005, in part by the Generalitat Valenciana under Contract APOSTD/2017/007, in part by the Spanish Ministry of Economy under Project ESP2016-79503-C2-2-P and Project TIN2015-63646-C5-5-R, in part by the National Natural Science Foundation of China under Grant 61771496, in part by the National Key Research and Development Program of China under Grant 2017YFB0502900, and in part by the Guangdong Provincial Natural Science Foundation under Grant 2016A030313254. *(Corresponding author: Jun Li.)*

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, 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, Universitat Jaume I, 12071 Castellón de la Plana, Spain (e-mail: rufernan@uji.es; pla@uji.es).

J. Li is with the Guangdong Provincial Key Laboratory of Urbanization and Geosimulation, Center of Integrated Geographic Information Analysis, School of Geography and Planning, Sun Yat-sen University, Guangzhou 510275, China (e-mail: lijun48@mail.sysu.edu.cn).

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

Digital Object Identifier 10.1109/TGRS.2018.2871782

larly relevant performance over HSI data due to their potential to effectively characterize spectral–spatial features [18], [19]. From regular stacked autoencoders (SAEs) [16], through sparse autoencoders [20], to deep belief networks [21], several kinds of deep learning models have been proposed and successfully adopted to classify HSI data. However, the 2-D 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) [22]. Precisely, the most recent approaches try to relieve this constraint by managing the HSI data as a whole 3-D volume in order to capture features representing the spectral–spatial domain. For instance, this is the case of the spatial-updated deep autoencoder presented in [23], 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.* [24] to classify remotely sensed HSI data, achieving the current state-of-the-art performance.

Since Chen *et al.* [24] adopted the CNN approach for HSI classification purposes, different CNN-based extensions have also been proposed in the literature to learn enhanced spectral–spatial features. For instance, Li *et al.* [25] propose 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 [26] also propose a classification approach that 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] and [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 [29] 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 a 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 present 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 [24], [30]. However, this method fails to capture local equivariances in the data and does not ensure equivariance at every layer within the CNN [31].

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 [32], 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 [33]–[35] (ResNet) and dense skip connections [36], [37] (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 [38]), 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 by Sabour *et al.* [39], where they introduced the concept of capsule networks (CapsNets) to encode the data relationships into an activity vector (rather than a scalar) whose length and orientation represent 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 [39] 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 data sets, 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 this paper is organized as follows. Section II discusses some advantages and limitations of CNNs for HSI classification which 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-known HSI data sets. Finally, Section V concludes this paper with some remarks and hints at plausible future research lines.

II. Advantages and Limitations of CNNs for HSI Classification

Let us denote by $\mathbf{X} \in \mathbb{R}^{H\times W\times C}$ an 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 $i = 1, 2, \ldots, H$, $j = 1, 2, \ldots, W$ and $k = 1, 2, \ldots, H \cdot W$. 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 pixelwise methods that understand the HSI data cube as a list of spectral vectors, for which they define complex, nonlinear 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 [40].

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 [29], [41], [42]. 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 data set $\mathcal{D}_{\text{train}} = \{\mathbf{x}_t, y_t\}_{t=1}^{N_{\text{labeled}}}$, 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, which often results in characteristic interferences in the obtained classification results (see Fig. 1). Specifically, the CNN model can work as a traditional pixelwise method, taking each pixel $\mathbf{x}_t$ as an input feature and applying spectral processing (i.e., the so-called 1-D CNN model [24], [29], [42]). However, the 1-D CNN cannot always manage the complexity of spectral features, introducing "salt and pepper" noise in the obtained classification [see Fig. 1 (left)]. In this sense, it is desirable to incorporate spatial information, i.e., by processing the 2-D regions of $\mathbf{X}$, usually centered on pixel $\mathbf{x}_{i,j}$, as input features (i.e., the 2-D 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 3-D CNN model) can reduce the intraclass 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: 1) the feature extractor net, composed by a stack of layers of artificial neurons (i.e., a convolutional layer followed by a nonlinear function and, often, by a subsampling or pooling layer) and 2) 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.

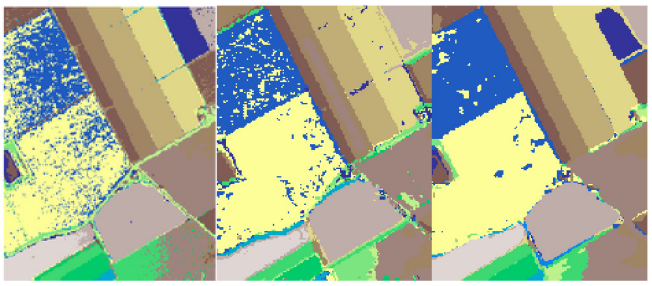


Fig. 1. Characteristic introduced in the classification results obtained by CNN models. Here, we show the examples of (Left) "salt and pepper" noise (1-D CNN), (Center) misclassified patches (2-D-CNN), and (Right) mixed regions (3-D CNN). The examples correspond to an area of an HSI scene collected over the SV in California, which will be described in detail in Section IV.

Focusing on the feature extractor net, the convolutional layer is the key block of the CNN. Instead of feedforward neural networks such as the MLP, where the group of neurons that compose the lth layer is fully connected with the neurons of the $l-1$th and $l+1$th layers, the lth 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 [43], 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 lth 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. In addition, this architecture assumes that elements (such as pixels in an 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 intraclass variability due to a better characterization of contextual features. In essence, each kernel of the lth 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:

$$o_{i,j,t}^{(l)z} = (\mathbf{X}^{(l)} * \mathbf{W}^{(l)})_{i,j,t} = \sum_{\hat{i}=0}^{k-1}\sum_{\hat{j}=0}^{k-1}\sum_{\hat{t}=0}^{q-1} x_{(i\cdot s+\hat{i}),(j\cdot s+\hat{j}),(t\cdot s+\hat{t})}^{(l)} \cdot w_{\hat{i},\hat{j},\hat{t}}^{(l)} + b^{(l)} \quad (1)$$

where $o_{i,j,t}^{(l)z}$ corresponds to the (i, j, t) element of the zth feature map that composes the output volume $\mathbf{O}^{(l)}$ of the lth convolutional layer, $x_{i,j,t}^{(l)}$ is the (i, j, t) element of the input volume $\mathbf{X}^{(l)}$, $w_{\hat{i},\hat{j},\hat{t}}^{(l)}$ 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-D 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:

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

where γ and β are learnable parameter vectors and ϵ is a parameter for numerical stability.

As convolution layers define a linear operation of elementwise 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 nonlinear activation function $\mathbf{O}^{(l)} = \mathrm{f}\left(\mathbf{O}^{(l)}\right)$, where $\mathrm{f}(\cdot)$ defines an elementwise function such as the sigmoid, the tanh, or the widely used rectified linear unit (ReLU) [44]–[46], which computes $\mathrm{f}(\mathbf{O}^{(l)}) = \max(0, \mathbf{O}^{(l)})$, 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 compared with 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 nonlinear 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 [47], stochastic pooling [48], or wavelet pooling [49]) over a neighborhood window [50]. Nonlinear 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 right 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, 2-D CNNs and 3-D CNNs, which indicate 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 [39], 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 paper 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. In addition, in order to provide accurate classification results, our proposal exploits both the spectral and the spatial

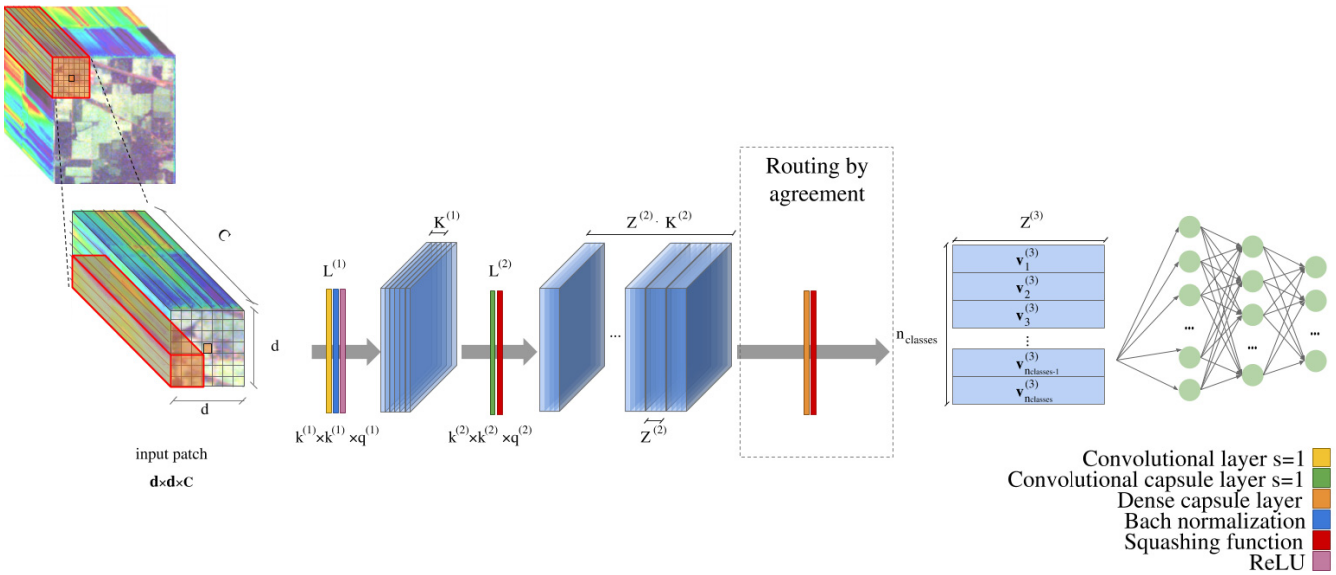


Fig. 2. Proposed neural network architecture. The neural model is composed by an encoder network (in blue) and a decoder network (in green).

information contained into the data cube $\mathbf{X}$, implementing a 3-D 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 an 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, 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 3-D neighboring blocks around each $\mathbf{x}_{i,j}$ (called patches and denoted by $\mathbf{p}_{i,j} \in \mathbb{R}^{d \times d \times C}$), where $d \times d$ is the size of the spatial patch and C is 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 the input data, following a border mirroring strategy described in detail in [29].

The proposed architecture is shown in Fig. 2, where two main parts are clearly differentiated. The HSI data introduced into the model are 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 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 mth capsule will apply its $Z^{(2)}$ units over a region of the volume $\mathbf{O}^{(1)}$, obtaining as a result the output vector $\mathbf{u}_m^{(2)} \in \mathbb{R}^{Z^{(2)}} = [u_{m,1}^{(2)}, u_{m,2}^{(2)}, \ldots, u_{m,Z^{(2)}}^{(2)}]$. 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 [51]. 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

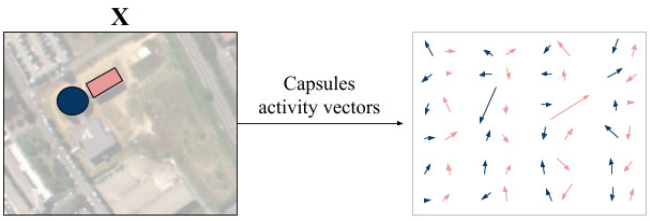


Fig. 3. Given an input image $\mathbf{X}$ with several objects, such as buildings with different shapes, the output of each capsule will be an activity vector whose length and orientation give the likelihood of the object and its instantiation parameters. In this sense, each capsule is in charge of finding 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 UP 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.

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 multidimensional entities present 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 vector obtained by each capsule of dimension $Z^{(2)}$. An important aspect is that these groups of neurons allow the mth 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 mth 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 nonlinear squashing function expressed by (3), which can be understood as the nonlinear 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

$$\tilde{\mathbf{u}}_m^{(l)} = \frac{\left\|\mathbf{u}_m^{(l)}\right\|^2}{1+\left\|\mathbf{u}_m^{(l)}\right\|^2} \cdot \frac{\mathbf{u}_m^{(l)}}{\left\|\mathbf{u}_m^{(l)}\right\|}. \tag{3}$$

3) Third Layer: After computing the outputs of the primary capsule layer and applying the nonlinear squashing function of (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* [39], 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 nth capsule in the current layer $L^{(l)}$ takes as the input data all the output vectors of the $K^{(l-1)}$ capsules located at the previous layer $L^{(l-1)}$, obtaining for each one a prediction vector $\hat{\mathbf{u}}_m^{(l)}$, with $m = 1, 2, \ldots, K^{(l-1)}$, calculated as the weighted multiplication between the mth 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 mth capsule in layer $L^{(l-1)}$ with the nth capsule in layer $L^{(l)}$, as shown in the following equations:

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

where $B_n^{(l)}$ are the biases of capsule n. This equation can be interpreted as a transformation where the output volume from the 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 mth capsule in layer $L^{(l-1)}$ and the nth capsule in layer $L^{(l)}$.

Moreover, the obtained prediction vectors can be interpreted as the vote of each capsule of $L^{(l-1)}$ in the output of the nth 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 outputs 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 (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

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

Focusing again on coupling coefficients, $c_{m,n}^{(l)}$ measures the probability that capsule m activates capsule n, and 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

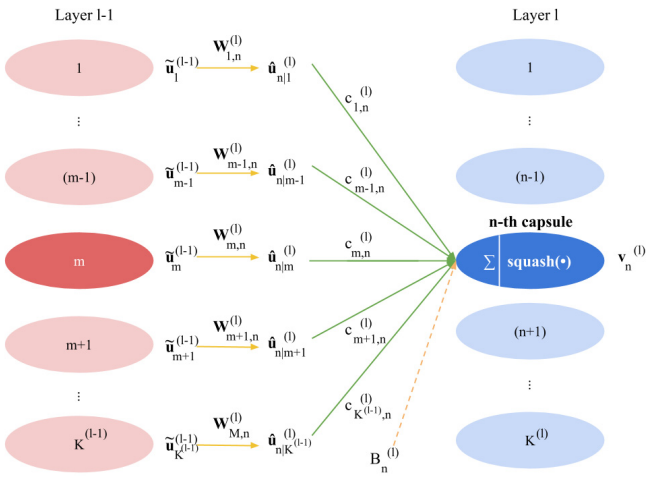


Fig. 4. Dynamic routing between capsules: the inferior-layer capsule activity vector is the current input vector $\tilde{\mathbf{u}}_m^{(l-1)}$ of the higher layer capsule n. After a matrix transformation given by (4), $\tilde{\mathbf{u}}_m^{(l-1)}$ is transformed into the prediction vector $\hat{\mathbf{u}}_{n|m}^{(l)}$. The weighted sum [see (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 (6), gives the nth capsule activity vector $\mathbf{v}_n^{(l)}$.

obtained by the *routing softmax* expressed by the following equation:

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

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:

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

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 $\cos(\theta) = \cos(0) = 1$, and 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–whole 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 shows 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 $i = 1, 2, \ldots, n_{\text{classes}}$). 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 $\|\mathbf{v}_i^{(l)}\|$ the probability of belonging to class i. The goodness of the network's output with regard to the desired output can be calculated by the loss function

$$\begin{aligned} L_{\text{margin}} = \sum_i^{n_{\text{classes}}} &\left(T_i \max\left(0, \alpha^+ - \left\|\mathbf{v}_i^{(l)}\right\|\right)^2\right. \\ &\left. + \lambda(1 - T_i)\max\left(0, \left\|\mathbf{v}_i^{(l)}\right\| - \alpha^-\right)^2\right) \end{aligned} \tag{9}$$

where T_i is set to 1 if class i is present in the data and 0 otherwise. We can observe two well-differentiated parts (addends) in (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 $T_i = 0$). In addition, parameters α^+ and α^- work as boundaries, forcing the length of the activity vector $\|\mathbf{v}_i^{(l)}\|$ (i.e., the probability) in (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. This expression can be extended in order to improve the final classification accuracy by adding a typical reconstruction loss $L_{\text{recon}} = \|\mathbf{X} - \mathbf{X}'\|$, 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, with the aim of improving the fine-tuning process of the parameters employed in the proposed network.

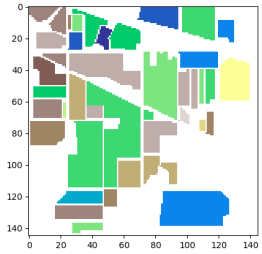
B. Decoder Network

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 (10) employing the Adam optimizer [52] with a learning rate equal to 0.001 and 100 training epochs

$$L_{\text{final}} = L_{\text{margin}} + \theta L_{\text{recon}} \tag{10}$$

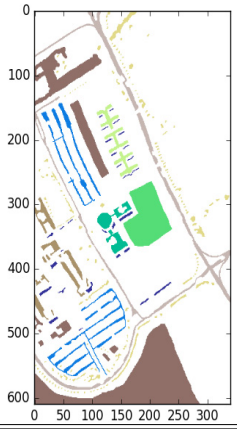
where θ is a regularization factor to balance the weight between both loss measures that has been fixed to $\theta = 0.0005 \cdot C$ after a grid search in order to assign an appropriate weight to the reconstruction loss. Finally, Table I summarizes the layers that compose the proposed model, indicating their configuration parameters, which have been demonstrated a good performance with tested HSI data sets.

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)

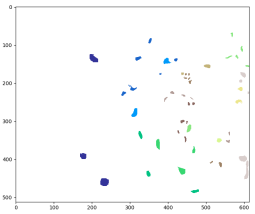


Color	Land-cover type	Samples
	Background	164624
	Asphalt	6631
	Meadows	18649
	Gravel	2099
	Trees	3064
	Painted metal sheets	1345
	Bare Soil	5029
	Bitumen	1330
	Self-Blocking Bricks	3682
	Shadows	947
	Total samples	207400

SALINAS VALLEY (SV)

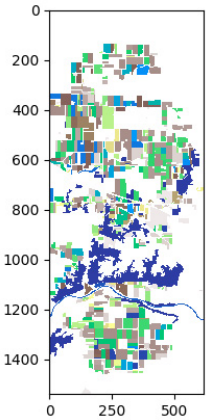
Color	Land-cover type	Samples
	Background	56975
	Brocoli-green-weeds-1	2009
	Brocoli-green-weeds-2	3726
	Fallow	1976
	Fallow-rough-plow	1394
	Fallow-smooth	2678
	Stubble	3959
	Celery	3579
	Grapes-untrained	11271
	Soil-vinyard-develop	6203
	Corn-senesced-green-weeds	3278
	Lettuce-romaine-4wk	1068
	Lettuce-romaine-5wk	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

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				

Fig. 5. Number of available samples in the IP, UP, and SV HSI data sets.

IV. Experimental Results

A. Hyperspectral Data Sets

Five real hyperspectral data sets have been considered in our experiments (see Fig. 5). These are the Indian Pines (IP), Salinas Valley (SV), Kennedy Space Center (KSC), and the full version of the IP scene, referred hereinafter as the big IP scene (BIP), all captured by the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) sensor [53], and the University of Pavia (UP) image, acquired by the Reflective Optics System Imaging Spectrometer (ROSIS) sensor [54]. In the following, we provide a description of the aforementioned data sets.

1) *IP:* The IP data set 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 a spatial resolution of 20 m/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.

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	

The IP data set contains a total of 16 mutually exclusive ground-truth classes.

2) *SV:* The SV image was captured in 1998 by the AVIRIS sensor over the SV, CA, USA. The data comprise 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 classes are included in the SV ground-truth data.
3) *KSC:* The KSC image was also collected by the AVIRIS instrument (1996) over the KSC 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) *UP:* The UP data set was gathered by the ROSIS sensor (in 2001) over the UP, 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 nine different class labels.
5) *BIP:* The BIP image comprises the full flight line of the IP data set captured by the AVIRIS sensor in 1992. This image contains 2678×614 pixels (20 mpp) 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 V. This data set 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 eight different classification methods have been selected to conduct the experimental validation in this paper. Specifically, the SVM with radial basis function kernel [55], the RF classifier, the MLP as well as a deep MLP version with four layers, the 2-D CNN, the 3-D CNN [24], the spectral–spatial residual network (SSRN) [34], and the deep fast CNN (DFCNN) [29] have been compared with the proposed approach. Note that the SVM, RF, and MLP are spectral classifiers, while the 2-D CNN is a spatial-based technique and the SSRN and DFCNN (together with the proposed approach) are all spectral–spatial methods. In the case of the 2-D CNN, the PCA has been used to reduce the number of HSI bands to a single principal component. In addition, all the hyperparameters 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 i7-6700K processor with 8M of Cache and up to 4.20 GHz (four cores/eight way multitask processing), 40 GB of DDR4 RAM with a serial speed of 2400 MHz, an NVIDIA GeForce GTX 1080 GPU with 8-GB GDDR5X of video memory and 10 Gb/s of memory frequency, a Toshiba DT01ACA HDD with 7200 RPM and 2 TB of capacity, and an ASUS Z170 pro-gaming motherboard. Regarding our software environment, it is composed by Ubuntu 16.04.4 x64 as an operating system, CUDA 9 and cuDNN 7.0.5, PyTorch framework [56], and Python 3.5.2 as the programing 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 data sets, considering the SVM, RF, MLP, 2-D CNN, and 3-D CNN classifiers together with the proposed approach. In Tables II–V, 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 2-D CNN, 3-D CNN, and the proposed method. It should also be mentioned that each table contains the corresponding average and standard deviation values after five Monte Carlo runs.

TABLE II

CLASSIFICATION RESULTS FOR THE IP DATA SET 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 UP DATA SET 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

TABLE IV

CLASSIFICATION RESULTS FOR THE SV DATA SET 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

TABLE V

CLASSIFICATION RESULTS FOR THE BIP DATA SET 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

From the results reported in Tables II–V, it is possible to observe that the proposed approach reaches a consistent performance improvement with respect to SVM, RF, MLP, 2-D CNN, and 3-D CNN classification methods, in global sense and also for the individual classes of the IP, UP, SV, and BIP data sets. Among all the competitors considered in this initial experiment, the spectral–spatial classifier 3-D 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 2-D CNN classifiers. Nonetheless, the proposed approach is able to consistently outperform the 3-D CNN with an average improvement of +1.93, +3.84, and +2.46 for OA, AA, and kappa metrics, respectively. Among all these quantitative results, the experimental comparison conducted over the BIP scene deserves special attention because of the complexity of this data set. 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 in 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 consider the spectral information contained in the HSI data. In addition, the 2-D 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

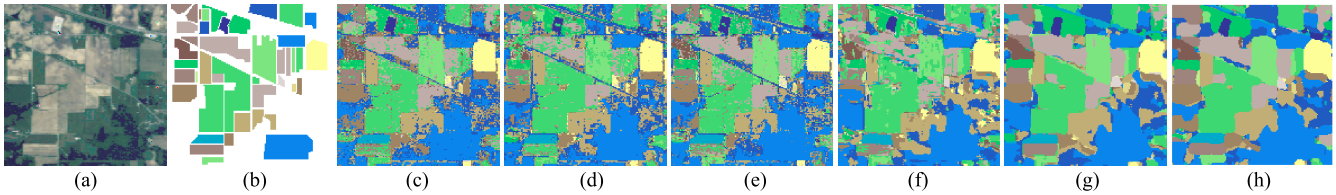


Fig. 6. Classification maps for the IP data set. (a) Simulated RGB composition of the scene. (b) Ground-truth classification map. (c)–(h) Classification maps corresponding to Table II [SVM (86.24%), RF (78.55%), MLP (85.27%), 2-D CNN (83.59%), 3-D CNN (97.81%), and proposed (99.45%), respectively]. Note that the overall classification accuracies are shown in brackets and the best result is highlighted in bold font.

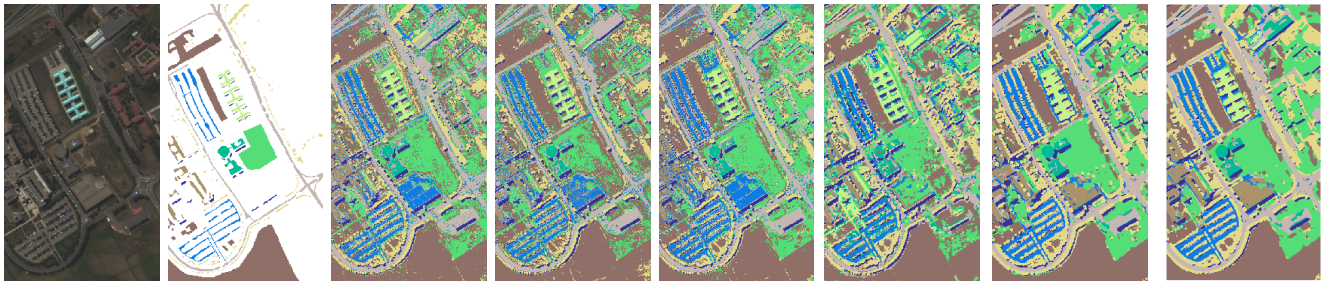


Fig. 7. Classification maps for the UP data set. (a) Simulated RGB composition of the scene. (b) Ground-truth classification map. (c)–(h) Classification maps corresponding to Table III [SVM (95.20%), RF (92.03%), MLP (94.82%), 2-D CNN (94.77%), 3-D CNN (98.54%), and proposed (99.95%), respectively]. Note that the overall classification accuracies are shown in brackets and the best result is highlighted in bold font.

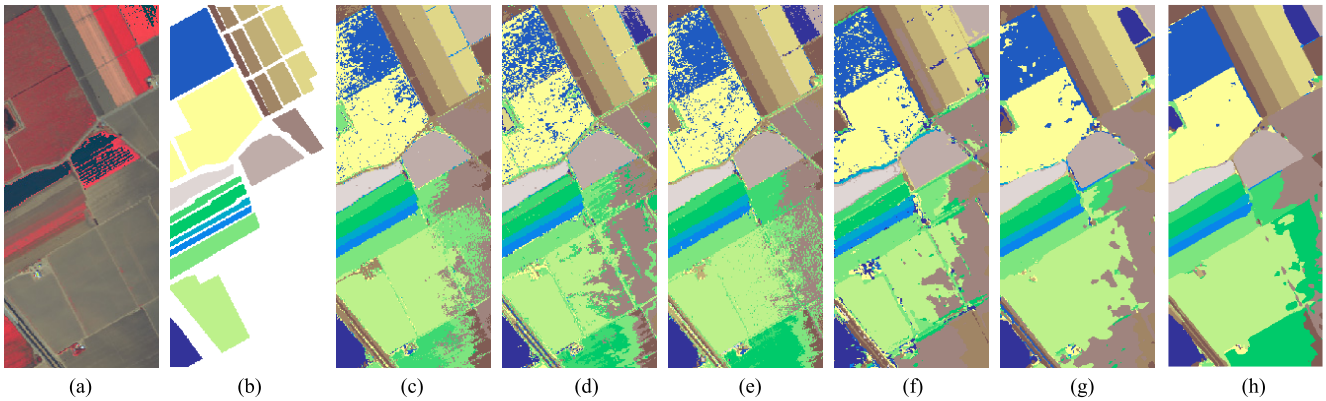


Fig. 8. Classification maps for the SV data set. (a) Simulated RGB composition of the scene. (b) Ground-truth classification map. (c)–(h) Classification maps corresponding to Table IV [SVM (94.15%), RF (90.76%), MLP (93.87%), 2-D CNN (92.31%), 3-D CNN (97.44%), and proposed (99.81%), respectively]. Note that the overall classification accuracies are shown in brackets and the best result is highlighted in bold font.

observe that the 3-D CNN generates better results than the SVM, RF, MLP, and 2-D CNN in terms of class consistency. However, the proposed approach produces better results in terms of border delineation and OA. For instance, 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 3-D 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 [34] and DFCNN [29]. 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 data sets. Note that the tested spatial sizes are presented in rows and the considered data sets are arranged in columns to show the average OA result and also the corresponding standard deviation in brackets (after five 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.

TABLE VI

OVERALL ACCURACY (%) ACHIEVED BY THE SSRN METHOD [34] 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 3-D CNN [24], DFCNN [29], AND THE PROPOSED APPROACH WITH THE IP DATA SET 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**

From the results reported in Table VI, the proposed network architecture consistently outperforms the SSRN for most tested configurations. More specifically, the average OA 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, and +0.23 for the IP, KSC, and UP data sets, 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 efficient 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.

In addition, Tables VII and VIII give an experimental comparison among the 3-D CNN [24], DFCNN, [29] and the proposed approach using the IP and UP data sets 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 3-D CNN, DFCNN, and the proposed approach, respectively, with different spatial sizes.

Some important observations can be made from Tables VII and 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 also be 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 data set, 3-D 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 also be observed in the experiments with the UP data set. This suggests that the proposed approach is able to uncover more descriptive features than the 3-D CNN and DFCNN techniques.

For illustrative purposes, Fig. 9 shows the classification maps obtained by the DFCNN [29] and the proposed approach

TABLE VIII

QUANTITATIVE COMPARISON OF THE 3-D CNN [24], DFCNN [29], AND THE PROPOSED APPROACH WITH THE UP DATA SET 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**

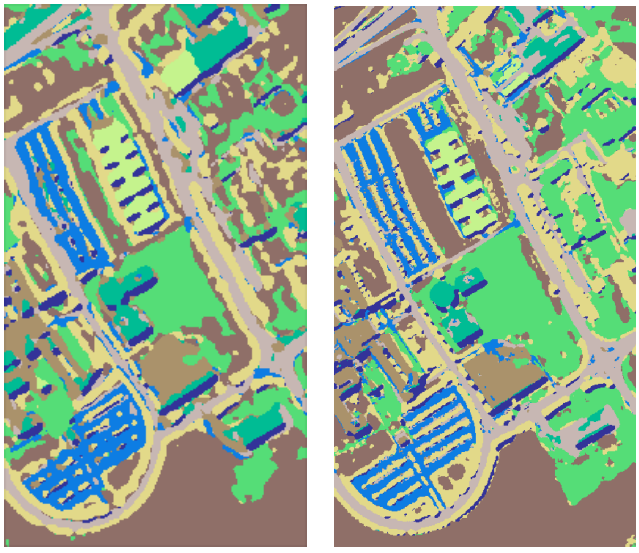


Fig. 9. Classification maps obtained by (Left) DFCNN [29] and (Right) proposed approach for the UP data set. A visual comparison of both maps indicates that the proposed method provides better class delineation and definition of urban features, for instance, in classes such as self-blocking bricks (blue) or bitumen (dark green), containing both circular and rectangular urban features.

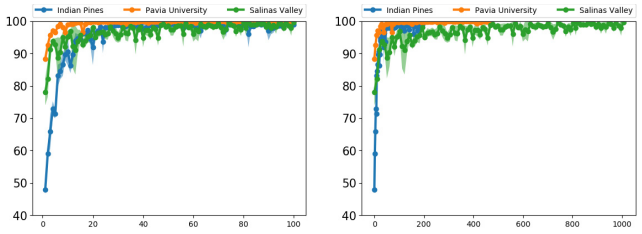


Fig. 10. Evolution of the test accuracy (in %) of (Left) proposed approach (y-axis) versus epochs and (Right) computational time in seconds for the experiments with the IP, UP, and SV data sets.

for the UP data set. 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. In addition, the classification results obtained by the proposed approach over unlabeled image areas appear 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) and computational time in seconds (right). 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 the 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., 3-D CNN [24], SSRN [34], and DFCNN [29] (see Tables VI–VIII and Figs. 9 and 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 3-D 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 overfitting phenomenon inherent to complex and deep networks.

V. Conclusion

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 that 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 paper, which consider five well-known HSI data sets and eight 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 data sets 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.

References

[1] D. Landgrebe, "Hyperspectral image data analysis," *IEEE Signal Process. Mag.*, vol. 19, no. 1, pp. 17–28, Jan. 2002.

[2] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. M. 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.

[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 Process. Mag.*, vol. 31, no. 1, pp. 45–54, Jan. 2014.

[4] A. Ghiyamat and H. Z. Shafri, "A review on hyperspectral remote sensing for homogeneous and heterogeneous forest biodiversity assessment," *Int. J. Remote Sens.*, 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," *Proc. IEEE*, vol. 101, no. 3, pp. 652–675, Mar. 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 J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 9, no. 9, pp. 4117–4128, Sep. 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 Sens. Lett.*, 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 *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 233–242.

[9] E. G. Njoku, *Encyclopedia of Remote Sensing*. New York, NY, USA: Springer, 2014.

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

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

[12] J. Haut, M. Paoletti, J. Plaza, and A. Plaza, "Cloud implementation of the K-means algorithm for hyperspectral image analysis," *J. Supercomput.*, vol. 73, no. 1, pp. 514–529, 2017.

[13] Y. Bazi and F. Melgani, "Gaussian process approach to remote sensing image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 1, pp. 186–197, Jan. 2010.

[14] 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.

[15] J. M. Haut, M. E. Paoletti, J. Plaza, and A. Plaza, "Fast dimensionality reduction and classification of hyperspectral images with extreme learning machines," *J. Real-Time Image Process.*, pp. 1–24, 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 J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 6, pp. 2094–2107, Jun. 2014.

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

[18] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

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

[20] 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," *Eur. J. Remote Sens.*, vol. 50, no. 1, pp. 47–63, 2017.

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

[22] 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.

[23] 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, Sep. 2016.

[24] 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.

[25] 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.

[26] W. Shao 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, Oct. 2016.

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

[28] 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, 2017.

[29] 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.*, [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0924271617303660, doi: 10.1016/j.isprsjprs.2017.11.021.

[30] J. Acquarelli, E. Marchiori, L. M. C. Buydens, T. N. Tran, and T. van Laarhoven, "Spectral-spatial classification of hyperspectral images: Three tricks and a new supervised learning setting," *CoRR*, Nov. 2017. [Online]. Available: http://arxiv.org/abs/1711.05512

[31] D. E. Worrall, S. J. Garbin, D. Turmukhambetov, and G. J. Brostow, "Harmonic networks: Deep translation and rotation equivariance," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2017.

[32] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2377–2385.

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[34] 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, Feb. 2018.

[35] 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.*, to be published, doi: 10.1109/TGRS.2018.2860125.

[36] G. Huang, Z. Liu, L. Van Der Maaten, K. Q. Weinberger, "Densely Connected Convolutional Networks," in *Proc. CVPR*, vol. 1, no. 2, 2017, pp. 1–3,

[37] M. E. Paoletti, J. M. Haut, J. Plaza, and A. Plaza, "Deep&dense convolutional neural network for hyperspectral image classification," *Remote Sens.*, vol. 10, no. 9, p. 1454, 2018. [Online]. Available: http://www.mdpi.com/2072-4292/10/9/1454

[38] 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. (CVPR)*, Jul. 2017, pp. 5987–5995.

[39] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 3859–3869.

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

[41] P. Fisher, "The pixel: A snare and a delusion," *Int. J. Remote Sens.*, vol. 18, no. 3, pp. 679–685, 1997.

[42] 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.*, to be published, doi: 10.1109/TGRS.2018.2838665.

[43] D. J. Field, "Wavelets, vision and the statistics of natural scenes," *Phil. Trans. Roy. Soc. London A, Math., Phys. Eng. Sci.*, vol. 357, no. 1760, pp. 2527–2542, 1999.

[44] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?" in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Oct. 2009, pp. 2146–2153.

[45] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, J. Fürnkranz and T. Joachims, Eds. Madison, WI, USA: Omni Press, 2010, pp. 807–814.

[46] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, G. J. Gordon and D. B. Dunson, Eds. 2011, pp. 315–323.

[47] D. Yu, H. Wang, P. Chen, and Z. Wei, "Mixed pooling for convolutional neural networks," in *Rough Sets Knowl. Technol.*, D. Miao, W. Pedrycz, D. Ślęzak, G. Peters, Q. Hu, and R. Wang, Eds. Cham, Switzerland: Springer, 2014, pp. 364–375.

[48] M. D. Zeiler and R. Fergus. (2013). "Stochastic pooling for regularization of deep convolutional neural networks." [Online]. Available: https://arxiv.org/abs/1301.3557

[49] T. Williams and R. Li, "Wavelet pooling for convolutional neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2018. [Online]. Available: https://openreview.net/forum?id=rkhlb8lCZ

[50] Y. T. Zhou and R. Chellappa, "Computation of optical flow using a neural network," in *Proc. IEEE Int. Conf. Neural Netw.*, vol. 2, Jul. 1988, pp. 71–78.

[51] G. E. Hinton, S. Sabour, and N. Frosst, "Matrix capsules with EM routing," in *Proc. Int. Conf. Learn. Represent.*, 2018. [Online]. Available: https://openreview.net/forum?id=HJWLfGWRb
[52] D. P. Kingma and J. Ba. (2014). "Adam: A method for stochastic optimization." [Online]. Available: https://arxiv.org/abs/1412.6980
[53] R. O. Green *et al.*, "Imaging spectroscopy and the airborne visible/infrared imaging spectrometer (AVIRIS)," *Remote Sens. Environ.*, vol. 65, no. 3, pp. 227–248, Sep. 1998.
[54] 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 *Optoelectronic technologies for remote sensing from space*, vol. 868. International Society for Optics and Photonics, 1988, pp. 134–142.
[55] B. Waske, S. van der Linden, J. Benediktsson, A. Rabe, and P. Hostert, "Sensitivity of support vector machines to random feature selection in classification of hyperspectral data," *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 7, pp. 2880–2889, Jul. 2010.
[56] A. Paszke *et al.* (2017). *Automatic Differentiation in PyTorch*. [Online]. Available: https://openreview.net/forum?id=BJJsrmfCZ and https://nips.cc/Conferences/2017/Schedule?showEvent=8779

Mercedes E. Paoletti (S'17) received the B.Sc. and M.Sc. degrees in computer engineering from the University of Extremadura, Cáceres, Spain, in 2014 and 2016, respectively, where she is currently pursuing the Ph.D. degree under the University Teacher Training Programme from the Spanish Ministry of Education.

She is currently a member of the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, University of Extremadura. Her research interests include remote sensing and analysis of very high spectral resolution with the current focus on deep learning and high-performance computing.

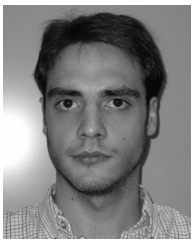
Juan Mario Haut (S'17) received the B.Sc. and M.Sc. degrees in computer engineering from the University of Extremadura, Cáceres, Spain, in 2011 and 2014, respectively, where he is currently pursuing the Ph.D. degree under the University Teacher Training Programme from the Spanish Ministry of Education.

He is currently a member of the Hyperspectral Computing Laboratory, Department of Computers and Communications, University of Extremadura, His research interests include remote sensing and analysis of very high spectral resolution with the current focus on deep learning and cloud computing. https://mhaut.github.io/.

Ruben Fernandez-Beltran received the B.Sc. degree in computer science, the M.Sc. degee in intelligent systems, and the Ph.D. degree in computer science from Universitat Jaume I, Castellón de la Plana, Spain, in 2007, 2011, and 2016, respectively.

He has been Visiting Scientist with the University of Bristol, Bristol, U.K. Since 2017, he has been a Visiting Post-Doctoral Researcher with the Hyperspectral Computing Laboratory, University of Extremadura, Cáceres, Spain. He is currently a Post-Doctoral Researcher with the Computer Vision Group, Universitat Jaume I, where he is also a member of the Institute of New Imaging Technologies. His research interests include multimedia retrieval, spatiospectral image analysis, and pattern recognition techniques applied to image processing and remote sensing.

Dr. Fernandez-Beltran is a member of the Spanish Association for Pattern Recognition and Image Analysis, which is part of the International Association for Pattern Recognition. He received the Outstanding Ph.D. Dissertation Award from Universitat Jaume I in 2017.



Javier Plaza (M'09–SM'15) received the M.Sc. and Ph.D. degrees in computer engineering from the Department of Technology of Computers and Communications, University of Extremadura, Cáceres, Spain, in 2004 and 2008, respectively.

He is currently a member of the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, University of Extremadura. He has authored over 150 publications, including over 50 JCR journal papers, 10 book chapters, and 90 peer-reviewed conference proceeding papers. His research interests include hyperspectral data processing and parallel computing of remote sensing data.

Dr. Plaza was a recipient of the Outstanding Ph.D. Dissertation Award from the University of Extremadura in 2008, the Most Highly Cited Paper Award from the *Journal of Parallel and Distributed Computing* from 2005 to 2010, and the Best Column Award of the *IEEE Signal Processing Magazine* in 2015. He received the best paper awards from the IEEE International Conference on Space Technology and the IEEE Symposium on Signal Processing and Information Technology. He has guest edited four special issues on hyperspectral remote sensing for different journals. He is currently an Associate Editor of the IEEE GEOSCIENCE AND REMOTE SENSING LETTERS and the IEEE Remote Sensing Code Library. http://www.umbc.edu/rssipl/people/jplaza.

Antonio Plaza (M'05–SM'07–F'15) received the M.Sc. and Ph.D. degrees in computer engineering from the Department of Technology of Computers and Communications, University of Extremadura, Cáceres, Spain, in 1999 and 2002, respectively.

He is currently the Head of the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, University of Extremadura. He has authored over 600 publications, including over 200 JCR journal papers (over 160 in IEEE journals), 23 book chapters, and around 300 peer-reviewed conference proceeding papers. His research interests include hyperspectral data processing and parallel computing of remote sensing data.

Dr. Plaza was a member of the Editorial Board of the *IEEE Geoscience and Remote Sensing Newsletter* from 2011 to 2012 and the *IEEE Geoscience and Remote Sensing Magazine* in 2013. He was also a member of the Steering Committee of the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING (JSTARS). He is a fellow of IEEE for the contributions to hyperspectral data processing and parallel computing of Earth observation data. He was a recipient of the Recognition of Best Reviewers of the IEEE GEOSCIENCE AND REMOTE SENSING LETTERS in 2009 and the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING in 2010, for which he served as an Associate Editor from 2007 to 2012. He was a recipient of the Most Highly Cited Paper Award from the *Journal of Parallel and Distributed Computing* from 2005 to 2010, the 2013 Best Paper Award of the JSTARS journal, and the Best Column Award of the *IEEE Signal Processing Magazine* in 2015. He received the Best Paper Awards at the IEEE International Conference on Space Technology and the IEEE Symposium on Signal Processing and Information Technology. He has guest edited 10 special issues on hyperspectral remote sensing for different journals. He served as the Director of Education Activities for the IEEE Geoscience and Remote Sensing Society (GRSS) from 2011 to 2012 and the President of the Spanish Chapter of the IEEE GRSS from 2012 to 2016. He has reviewed over 500 manuscripts for over 50 different journals. He served as the Editor-in-Chief for the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING from 2013 to 2017. He is an Associate Editor of the IEEE ACCESS (receiving a recognition as an outstanding Associate Editor of the journal in 2017). http://www.umbc.edu/rssipl/people/aplaza.

Jun Li (SM'16) was born in Loudi, Hunan, China, in 1982. She received the Engineering Degree in geographical information systems from Hunan Normal University, Changsha, China, in 2004, the M.Sc. degree in remote sensing and photogrammetry from Peking University, Beijing, China, in 2007, and the Ph.D. degree in electrical and computer engineering from the Instituto Superior Tecnico, Technical University of Lisbon, Lisbon, Portugal, in 2011.

From 2011 to 2012, she was a Post-Doctoral Researcher with the Department of Technology of Computers and Communications, University of Extremadura, Badajoz, Spain. She is currently a Professor with the School of Geography and Planning, Sun Yat-sen University, Guangzhou, China, where she also founded her own research group on hyperspectral image analysis in 2013. Since 2013, she has obtained several prestigious funding grants at the national and international level. She has published a total of 69 journal citation report papers, 48 conference international conference papers, and one book chapter. She has received a significant number of citations to her published works, with several papers distinguished as Highly Cited Papers in Thomson ReutersâŁ™ Web of Science–Essential Science Indicators. Her research interests include remotely sensed hyperspectral image analysis, signal processing, supervised/semisupervised learning, and active learning.

Dr. Li's students have also received important distinctions and awards at international conferences and symposia. She has served as a Guest Editor of a Special Issue in the prestigious PROCEEDINGS OF THE IEEE journal. She has also served as a Guest Editor of a Special Issue in the prestigious *ISPRS Journal of Photogrammetry and Remote Sensing* journal. He has been serving as an Associate Editor of the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING since 2014.

Filiberto Pla received the B.Sc. and Ph.D. degrees in physics from the Universitat de Valéncia, Valencia, Spain, in 1989 and 1993, respectively.

He is currently a Full Professor with the Departament de Llenguatges i Sistemes Informatics, Universitat Jaume I, Castellón de la Plana, Spain. He has been a Visiting Scientist with the Silsoe Research Institute, University of Surrey, Guildford, U.K., the University of Bristol, Bristol, U.K., CEMAGREF, Montpellier, France, the University of Genoa, Genoa, Italy, the Instituto Superior Técnico, Lisbon, Portugal, the Swiss Federal Institute of Technology, ETH Zurich, Zürich, Switzerland, the idiap Research Institute, Switzerland, the Delft University of Technology, Delft, The Netherlands, and the Mid Sweden University, Sweden. He has been the Director of the Institute of New Imaging Technologies, Universitat Jaume I. His research interests include color and spectral image analysis, visual motion analysis, 3-D image visualization, and pattern recognition techniques applied to image processing.

Dr. Pla is a member of the Spanish Association for Pattern Recognition and Image Analysis, which is part of the International Association for Pattern Recognition.