

# Multiple Attention-Guided Capsule Networks for Hyperspectral Image Classification

Mercedes E. Paoletti, *Senior Member, IEEE*, Sergio Moreno-Álvarez, and Juan M. Haut, *Senior Member, IEEE*

**Abstract**—The profound impact of deep learning and particularly of convolutional neural networks (CNNs) in automatic image processing has been decisive for the progress and evolution of remote sensing (RS) hyperspectral imaging (HSI) processing. Indeed, CNNs have stated themselves as the current state-of-art, reaching unparalleled results in HSI classification. However, most CNNs were designed for RGB images and their direct application to HSI data analysis could lead to non-optimal solutions. Moreover, CNNs perform classification based on the identification of specific features, neglecting the spatial-relationships between different features (i.e., their arrangement) due to pooling techniques. The capsule network (CapsNet) architecture is an attempt to overcome this drawback by nesting several neural layers within a capsule, connected by dynamic routing, both to identify not only the presence of a feature, but also its instantiation parameters, and to learn the relationships between different features. Although this mechanism improves the data representations, enhancing the classification of HSI data, it still acts as a black box, without control of the most relevant features for classification purposes. Indeed, important features could be discriminated. In this paper, a new multiple attention-guided CapsNet is proposed to improve feature processing for RS-HSIs classification, both to improve computational efficiency (in terms of parameters) and to increase accuracy. Hence, the most representatives visual parts of the images are identified using a detailed feature extractor coupled with attention mechanisms. Extensive experimental results have been obtained on five real datasets, demonstrating the great potential of the proposed method compared to other state-of-the-art classifiers.

**Index Terms**—CapsNet, CNN, feature, HSI, Attention.

## I. INTRODUCTION

**T**ECHNOLOGICAL advances in Earth Observation (EO) and remote sensing (RS) at both hardware and software levels have resulted in the development of sophisticated aerial and satellite sensing systems. In particular, improvements in imaging spectroscopy have led to the development of advanced hyperspectral imaging (HSI) sensors, known as spectrometers, which acquire imagery in hundreds of narrow, contiguous spectral bands by recording electromagnetic spectrum in the range from visible to short-wave infrared wavelengths

This work was supported by in part Junta de Extremadura FEDER under Grant GR18060 and Grant GR21040 and by in part by 2021 Leonardo Grant for Researchers and Cultural Creators, BBVA Foundation. The BBVA Foundation accepts no responsibility for the opinions, statements and contents included in the project and/or the results thereof, which are entirely the responsibility of the authors. (Corresponding author: Juan M. Haut)

Mercedes E. Paoletti is with the Department of Computer Architecture, Complutense University, Madrid, Spain (mpaolett@ucm.es),

Sergio Moreno-Álvarez and Juan M. Haut is with the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, Escuela Politécnica, University of Extremadura, 10003 Cáceres, Spain (email: smoreno@unex.es, juanmariohaut@unex.es).

Manuscript received April 19, 2005; revised September 17, 2014.

over large observed areas [1], [2]. Thus, HSI scenes compose large data cubes in which each pixel measures the absorption of solar radiation by the observed materials, creating a spectral signature along the bands, the shape of which will depend on the physical and chemical characteristics of the surface, such as its composition and roughness. As a result, each pixel is considered as a  $B$ -dimensional vector, where  $B$  is the number of bands/channels that will define the spectral signature, which is unique for each type of surface material. Furthermore, the spectral depth and resolution determine the spatial resolution due to the limitations of the spectrometers and their high data rate, so that normally the higher the compactness of the channels, the lower the spatial resolution, increasing the size of the surface captured in each pixel. This negatively affects the quality of the spectral signature, which is usually very mixed by different types of materials. Moreover, this drawback could be intensified by the noise introduced by the spectrometers. In this regard, sensor instruments have usually been designed seeking a trade-off between spatial and spectral resolutions, where a higher spatial resolution improves the object detection in an image, whilst a higher spectral resolution results in the representation of more features to differentiate more materials. Nevertheless, improvements in sensing instruments, coupled with some processing techniques to remove defects in the image, such as unmixing and super-resolution, denoising and pansharpening methods, are an important effort to correct these limitations, providing HSI datasets with more spatial details and better spectral quality [3]–[6].

The rich and abundant spatial-spectral information contained within a HSI scene is pretty useful and accurate for some research tasks and it has widely been used for a wide range of activities, such as hydrology [7], urban planning [8], contaminated soil restoration [9] or agriculture [10], among others [11]. In this context, pixel-wise classification methods has a major role in the analysis of HSI data, as they categorize each pixel in the scene by assigning it a label corresponding to a land-cover type [12]. Important efforts have been done to produce accurate and efficient unsupervised and supervised classifiers [13]–[17]. Nevertheless, the full exploitation of HSI information involves a number of challenges that must be faced in order to correctly process the data. The large spectral dimension of the data makes the classifiers more complex, as they must handle a large number of features to discriminate between the different classes. There is also a large spectral variability due to spectral mixing, with a significant intra-class variability problem coupled with unbalanced training data and an important lack of labelled samples. The exploitation of spatial information attempts to reduce the uncertainties introduced

by spectral information. However, traditional methods tend to reshape the spatial information into a vector in order to concatenate it to the spectral information or to employ kernel-based algorithms and complex filters based on non-intuitive hand-crafted features [18]–[22].

### A. Revolution begins with convolution

In this context, the great success of convolutional models for the simultaneous processing of spectral-spatial information contained in HSI scenes is noteworthy. Indeed, the convolutional neural network (CNN) [23], [24] have paved the way for a wide range of deep artificial neural networks (DANNs) for RS-HSI data processing [25], [26]. The major strength of this model lies in its extraordinary ability to automatically extract features from the data, which are refined along a hierarchical structure until an abstract representation suitable for the classification task is obtained. This is done through the kernel mechanism implemented by each convolution layer. In fact, the convolution kernel is defined as a set of local-connected and shared weights that work as a filters to determine the presence or absence of a particular feature in an input data.

Regarding 2D-convolutions, the  $l$ -th layer defines a set of filters  $\mathbf{W}^l$ , comprising  $K^l$  3D-arrays with size  $M \times M \times K^{l-1}$  which are applied over the input data. In this sense, the kernel application implies that  $K^l$  different filters will be locally applied over regions of size  $M \times M \times K^{l-1}$  of the input data. Indeed, if  $\mathbf{X}^{l-1} \in \mathbb{R}^{N \times N \times K^{l-1}}$  is defined as an input set of feature maps, where  $N \times N$  are the features and  $K^{l-1}$  the maps, the kernel will slice its  $K^l$  filters as  $M \times M \times K^{l-1}$  windows over  $\mathbf{X}^{l-1}$ , conducting the linear element-wise product defined by Eq. (1a), where  $\mathbf{b}^l$  is the bias:

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

$$x_{i,j,t}^l = \sum_{m=-M/2}^{M/2} \sum_{n=-M/2}^{M/2} \sum_{k=1}^{K^{l-1}} x_{i-m,j-n,k}^{l-1} \cdot w_{m,n,k}^l + b_k^l \quad (1b)$$

The resulting feature maps  $\mathbf{X}^l \in \mathbb{R}^{N \times N \times K^l}$  are obtained as the weighted sum of the input features by the weights of each filter plus the corresponding bias. Eq. (1b) gives the detailed formulation to obtain the  $(i, j)$ -th output feature at the  $t$ -th output map ( $\forall t \in [1, K^l]$ ), which is obtained as the conversion result from the multiband input data. Moreover, from Eq. (1b) can be deduced that the weights of a filter are shared by all locations of the output map. That means each filter produces an output feature map which determines the presence of a particular feature. Then, output feature maps are stacked together into  $\mathbf{X}^l$  in the same layer and sent to the following layers to extract more abstract features based on the detection of simpler ones.

The application of different convolution layers involves nonlinear activation functions [27] to model the non-linearity within the data, i.e., to learn complex non-linear features, and dimensional reduction of the data after each convolution not only to reduce the training time but also to provide invariance to certain geometric transformations of the data, in particular translation. Although this data downsampling can be

controlled with the stride and padding hyperparameters, it is quite common to include pooling layers to obtain “summaries” of  $P \times P$  sub-regions on each output map, where  $P$  defines the spatial dimension of the pooling.

### B. Spatial relations: the Achilles’ heel of CNNs

Current literature on HSI data classification abounds with convolution-based models. The first works followed the traditional pixel-by-pixel approach, for instance Gao *et al.* reshaped each pixel spectrum in 2D arrays to feed their 2DCNN model [28]. However, this strategy flattened the spatial map, neglecting the spatial distribution patterns. To overcome this drawback, the use of input patches composed of a central pixel to be classified and its neighbors was soon adopted. For instance, Yue *et al.* [29] defined a neighbourhood region around the target pixels as 2D input patches, applying principal component analysis (PCA) to reduce the number of dimensions whilst preserving the spatial information. Also, Zhao and Du performed a PCA to prepare the input patches of the network, combining at the end the obtained spatial features with the spectral ones to perform the classification [30]. More advanced models take advantage of the automatic feature extraction to process spectral-spatial information simultaneously. For instance the authors in [31] conducted an extensive study on HSI processing with spectral-spatial features by implementing a more efficient convolution architecture, and considering different input patch sizes. Furthermore, some models combines traditional filters and handcrafted features within the CNN architecture, such as the 2DCNN model proposed by He *et al.* [32], which combined covariance matrices within the deep architecture to improve the classification results. Also Boggavarapu *et al.* [33] and Praveen *et al.* [34] have extracted some prior-information through Gabor filters, which are then processed by a 3DCNN.

With the development of new deep learning (DL) techniques, some improvements have been implemented to overcome the limitations of CNNs when dealing with HSI data. For instance, when the CNN becomes too deep, it has to face the overfitting and vanishing gradient problems. On the one hand, the high variability of the HSI data and the lack of labelled samples prevent the model from properly covering the variations of the data, which both degrades learning procedure and over-adjusts the trainable parameters. On the other hand, the depth of the network can negatively affects the calculation of the gradient signal during the backward step, which tend rapidly to zero, fading out. As a result, learning is stuck. Some techniques deal with the lack of labelled samples [35]–[37], for instance the 3D Generative Adversarial Minority Oversampling (3D-GAMO) model [38] conducts and oversampling of those 3D-HSI patches belonging to minority classes with a weak representation caused by the lack of training data. Indeed, the 3D-GAMO generates new samples for those classes with few labelled samples applying a data augmenting procedure based on the noise extracted from existing class-specific samples. Furthermore, as an attempt to deal with overfitting problems, the Graph Convolutional Networks for Hyperspectral Image Classification (GCN) [39]–[42] was proposed as an emerging network architecture to

effectively model relations between data samples using a graph structure, where the spectral information is described using different perspectives combined by an end-to-end fusion network. On the other hand, focusing on the depth of the model, the degradation problem can be countered by skip connections, which introduces direct paths to pass both the data and the gradients during the forward and backward steps, respectively. In this regard, residual networks (ResNets) [43]–[45] introduces direct connections from the beginning to the end of a residual block, re-utilizing features from different levels, whilst densely connected networks (DenseNets) [46], [47] introduces dense connections that concatenates the data between different layers in a block.

The implementation of skips connections paved the way to more sophisticated networks. In particular, CNNs are criticized for their black-box behavior, in which convolution kernels consume a lot of resources learning repetitive and non-relevant features for the final classification, exacerbating overfitting problems. In this context, skip connections have been widely used to embed *feature-map attention* within convolution-based architectures, providing a new way to enhance the discrimination power on convolution features by implementing trainable attention mechanisms [48], [49]. The attention mechanism avoids costly processing of the whole image, as it only focuses on some specific regions, which are emphasized whilst the rest are blurred. This is usually done by providing a set of feature representations extracted from the original data, to which some selection procedure is applied to dynamically highlight the most salient features as needed [50], [51]. In this sense, deep networks computes a dynamic feature extraction mechanism based on soft-attention masks to concentrate on the relevant features, while disregarding others, as Eq. (2) indicates:

$$\tilde{\mathbf{X}}^l = \mathbf{X}^l \cdot \omega. \quad (2)$$

Usually, the attention mask  $\omega = \mathcal{G}(\mathbf{X}^l, \theta)$  is extracted from the input data  $\mathbf{X}^l$  through several data transformations  $\mathcal{G}$  implemented as a *mask-path* within the network architecture, and then directly applied through a skip connection to the input features to emphasize the most discriminatory ones, obtaining a refined output feature representation  $\tilde{\mathbf{X}}^l$ . A significant effort has been made to implement effective attention mechanisms [52]–[58], which has significantly improved the classification results.

Despite these great advances, CNN-based models are not robust enough to the orientation and pose of the data, as they do not learn some geometric transformations (such as rotations and shift movements [59]) and cannot model the spatial relationships between different features, such as size, perspective and orientation [60]. In fact, the CNN works by determining the existence of several abstract/complex features that are common to samples belonging to a class (usually with more emphasis on texture than on shapes [61]). Nevertheless, it is not able to determine the spatial relationships between different features, i.e. the arrangement between them, due to the application of pooling techniques. Although the pooling layer provides some invariance to certain spatial transformations, this process does not take into account other useful information

(such as rotations, shifts, scaling...), disregarding the spatial arrangement between features. This results in an important loss of spatial information. Moreover, as the CNN model goes deeper, its receptive field also increases gradually due to the hierarchically stacked kernels, and, therefore, pooling operations in deeper layers produces a greater information loss [62].

### C. Proposal: Capsule networks with attention mechanism

With the aim of overcoming the drawbacks regarding the spatial relationships modeling, capsule networks (CapsNets) have been proposed in the literature as an alternative to CNNs [60]. These networks encode the properties of extracted features (i.e., pose, orientation,...) in a vector by nesting several neurons with a "capsule". unit. This provides a greater description of the features. Moreover, CapsNets implement a dynamic routing algorithm (routing-by-agreement) to model the connections between capsules. To apply this routing, the network follows a tree-like structure where each node of the tree correspond to an active capsule. In this way, a voting procedure to select most appropriate features is constructed, where each capsule vote for a higher capsule state (parents). Parents aggregates the votes and update their states. This routing mechanism enables the model to learn spatial relationships between different features, since higher layer capsules will only be activated by certain features detected in lower layer capsules (Fig. 1). As a result, the CapsNet architecture reaches better accuracy results than standard CNNs when classifying HSI data [63]–[65].

Despite the great results obtained by CapsNets, these networks are affected by a major overfitting problem, due to the large number of parameters to be trained and the cost of resources spent on redundant features. In fact, much of the complexity of the model lies in its routing algorithm, where activity vectors from the lower capsules are routed to the upper capsules through a series of affine transformations and the application of coupling coefficients. Hence, the routing algorithm is computationally expensive and introduces a large number of instantiation parameters that increase the complexity of the model. This seriously impairs the accuracy of the CapsNet, degrading rapidly with few training samples or with high data variability. To overcome these drawbacks, this work identifies three lines of improvement:

- 1) Improving the initial data representation by refining initial convolutional features will produce better features.
- 2) Simplifying and optimising the process of building PrimaryCaps will improve the efficiency of the model, reducing the computational load.
- 3) Making routing between capsules at different levels more robust and efficient will enable the model to work more reliably when training with few data or with high variability, reducing both the number of parameters and the computational load.

Although there are some previous efforts focused on improving each of the three targets proposed above, these methods usually tend to increase the complexity of the model, introducing a significant trade-off between performance and

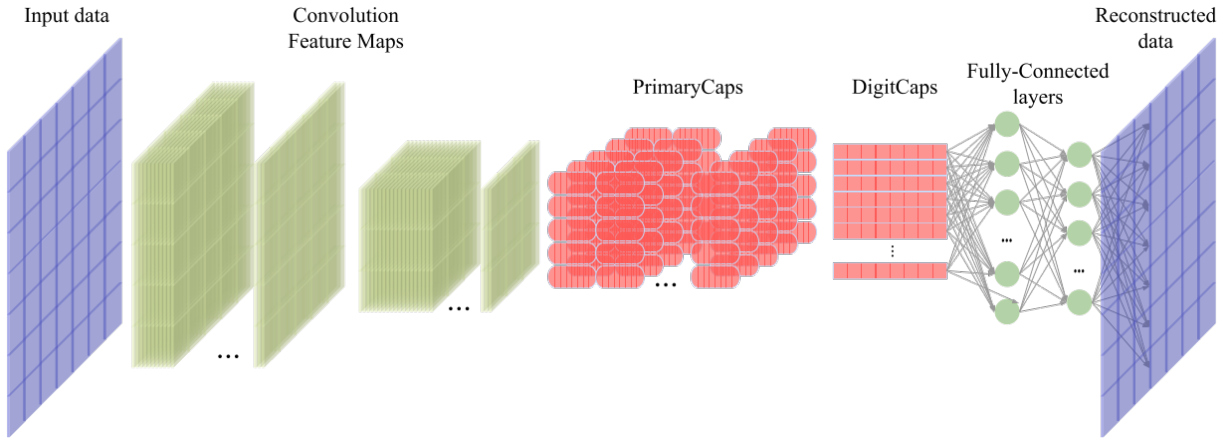


Fig. 1. Original CapsNet architecture. It is composed of one or several convolution layers, which extract the initial feature representations. These features are processed by the PrimaryCaps. Here, each capsule embeds several neurons, extracting an instantiation or activity vector for each feature. These outputs are dynamically routed to the DigitCaps, which determines for each class a vector whose content allows the final classification. Finally, to reinforce the processing, a reconstruction of the original data is performed using a series of fully connected layers.

complexity. In this context, and with the aim of addressing each target without hindering the model complexity, this work proposes a new implementation of a CapsNet for HSI data classification with multiple attention mechanisms [66], [67]. In particular, the proposed network implements three novel solutions:

- Regarding the first target, a soft-channel attention mask is extracted and applied within the input data in order to improve the initial feature data representation.

It should be noted that traditional channel attention mechanisms usually involve some computationally demanding operations. Indeed, to generate the channel weights several operational layers are introduced to capture non-linear cross-channel interactions, involving dimensionality reduction steps. Thus, both the model complexity and its computational load are appreciably aggravated.

To overcome this drawback, the proposed model introduced a novel and efficient channel-attention algorithm, which computes a local cross-channel interaction strategy to extract the relevant features. This is done through a light and fast 1D convolution, whose kernel size is adapted depending on the channel depth. This avoids the costly operations of dimensionality reduction and spectral-spatial recombinations. Moreover, the channel-attention forces the model to extract more refined features from the bottom layers, which will further enhance the subsequent feature extraction process.

- Regarding the second target, the proposed CapsNet implements a Depth-wise convolution layer before the PrimaryCaps layer. This layer works separately on each channel to combine spatial information. This greatly simplifies and reduces the number of parameters required during the capsule creation procedure. Furthermore, the capsules of the PrimaryCaps layer are obtained by a simple reshape of the features obtained by this layer.
- Regarding the third target, the proposed network implements a novel self-attention routing to connect PrimaryCaps with DigitCaps. As pointed before, traditional routing-by-agreement procedure introduces a large number of instan-

tiation parameters, increasing the complexity of the model and hampering its performance when few training data is available.

To overcome the drawbacks of traditional routing-by-agreement procedure, the proposed model implements a novel and highly parallelizable routing algorithm with self-attention mechanism to enhance the spatial relationship modeling whilst reducing the number of capsules. This routing capture long-range interactions between the data and guides the feature vectors of the PrimaryCaps to the DigitCaps more effectively. In this sense, the model changes the calculation of coupling coefficients, which are obtained through a self-attention tensor instead of log priors. This tensor stores the attention scores of the PrimaryCaps. Thus, the input of any capsule in the DigitCaps layer is implemented as the weighted sum of all the previous activity vectors, where the weight is obtained by self-attention. This provides a more robust and stable routing algorithm, while simplifying the calculation of the coefficients. Moreover, this mechanism prevents the explosion of a large number of instantiation parameters, which makes routing more efficient.

As a result, the proposed network can learn more descriptive feature representations, encoding rich information per feature, whilst take advantage of attention mechanism to cope with redundant features, enhancing the most discriminative ones and reducing the number of parameters. Furthermore, the implemented improvements reduce the complexity of the model, in terms of parameters, improving its computational efficiency.

The rest of the manuscript is organized as follows. Section II discusses previous related works. Section III provides the description and implementation details of the proposed network for HSI data classification. Section IV presents the results obtained after an exhaustive experimental analysis. Finally, section V concludes this paper with some remarks and future works.

## II. RELATED WORKS

CapsNets have been used for a wide range of applications as natural language processing [68], [69], sentiment analysis [70], image generation [71], medicine [72], object segmentation [73] or computer vision analysis [74]. In particular, there are some interesting works applying CapsNets for RS-HSI data classification. For instance, Zhang *et al.* [75] proposed a 1D-CapsNet to lighten the computational burden of traditional 3D-layers. Moreover, the authors in [63] proposed a HSI-CapsNet architecture to conduct spectral-spatial analysis. Deng *et al.* [64] implemented a modified two-layer CapsNet for HSI classification with limited training samples. Yin *et al.* [65] implemented a pre-training stage with three convolution layers that are transferred at the bottom of the CapsN architecture. Furthermore, Lei *et al.* [76] embedded an additional convolution layer before the capsule layers to capture high-level features and speed up the routing procedure. A sophisticated architecture was proposed by Wang *et al.* [77], where the CapsNet is combined with a triple generative adversarial network (TripleGAN) [78] to handle the spectral correlation, the high intra-class variability and inter-class similarity within HSI data. Also, inspired by dual-channel models, Jiang *et al.* [79] introduced two separate convolution channels before the capsule layers to extract spectral and spatial features separately and then concatenate them, whilst Wang *et al.* [80] exploited two GANs with capsules in a dual-channel architecture to efficiently model the relative position of samples.

These methods are inspired by the original works of Hilton *et al.* [81] and Sabour *et al.* [60]. In particular, the former ones introduced the capsule architecture to encode different feature properties (pose, orientation, scale, and illumination, for example) into a vector of instantiation parameters, whilst the latter ones implemented a dynamic and iterative mechanism to connect capsules of different levels, the routing-by-agreement, which sends those features extracted by low-level capsules to higher-level capsules depending on the probability of these features activating higher-level capsules.

Subsequent DL-based works have focused on enhancing the routing mechanism, in order to improve network performance in terms of accuracy and computational efficiency. Indeed, original routing-by-agreement is extremely expensive from a computational point of view and tend to produce poor results when dealing with complex datasets [82]–[85]. Several efforts have been done to overcome these drawbacks. For instance, Bahadori [86] introduced spectral capsules whose features are represented in an one-dimensional linear subspace to improve network convergence. Hinton *et al.* [87] introduced the Expectation-Maximization (EM) algorithm to improve the routing mechanism whilst Lenseen *et al.* [87] modified the EM to group similar predictions between matrix capsules. Wang and Liu [88] reinterpreted the routing mechanism as an optimization problem, implementing a clustering-based mechanism to improve it. Recently, Gu *et al.* [89] conducted a detailed investigation on the impact of routing on data processing with affine transformations, proposing a matrix transformation shared among all level capsules. Also, Zhang *et al.* [90] proposed a weighted kernel density estimation

framework to accelerate the routing procedure.

In this context, several works combine the capabilities of CapsNets with an attention-based mechanism, most of them as an addition to the original model, although there are some interesting works that introduce the mechanism within the routing to improve it between capsules of different levels. For instance, Zhang *et al.* [91] combined long short-term memory units (LSTMs) with attention mechanism and capsules for relation extraction in text/natural language analysis. Liu *et al.* [92] developed two components, i.e. a multi-element hierarchical attention module and a CapsNet for stock prediction, although attention is not included in the network itself but as part of the processing stack. Focusing in image processing, Choi *et al.* [93] introduces a new capsule activation function which is combined with attention routing through convolutional transforms. Huang and Zhou [94] introduced the attention after the first convolution layer and the after the primary capsule-layer (PrimaryCaps) of the architecture, the former through a standard global average pooling function and the latter implemented as a gate function via ReLU and hyperbolic tangent based on the output of the capsules. Hoogi *et al.* [95] incorporated the Self-Attention mechanism [96] between the convolution layer and the PrimaryCaps for image classification. On the other hand, Mazzia *et al.* [67] developed an efficient and light CapsNet by introducing the Self-Attention within the routing, embedding depth-wise separable convolutions within PrimaryCaps to reduce the amount of parameters [97]. Focusing on RS data processing, Ren *et al.* [98] proposed a deep dual-attention CapsNet inspired by the U-Net architecture and the channel feature attention for image segmentation.

Despite these efforts, none of them have been transferred to the analysis of RS-HSI data, the spectral and spatial complexity of which are particularly challenging. In fact, the literature does not report any capsule-based model that modifies the original routing by attention mechanism, thus, the current CapsNet-based models must cope with the limitations in terms of computational burden and overfitting. In this sense, this paper introduces a new deep capsule model for HSI data classification, which efficiently combines two attention procedures, the former at the beginning of the convolution-based feature extractor with the objective of refining the features that will be later processed by the capsules, whilst the latter includes self-attention algorithm to rout the low-level capsules to high-level ones. Details are provided in the following section

## III. METHODOLOGY

### A. Fundamentals of CapsNet

CapsNets were introduced to improve the representational capability of neural networks, encapsulating different properties of a feature in a vector instead of producing a single scalar [81]. This idea relies on the assumption that capsules and their routing mechanism encode the part-whole relationships between different entities (i.e. features, objects or object parts) into the instantiation vector, usually the position, shape, and existence of an entity. As a result, the capsules represent a

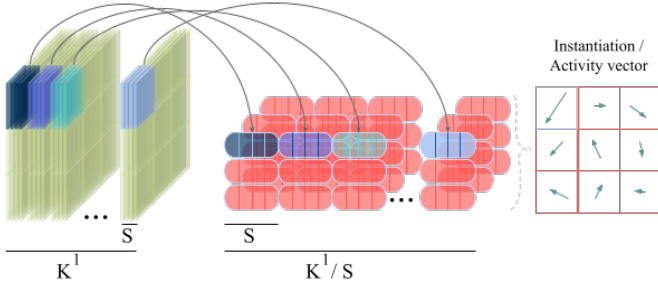


Fig. 2. The extracted low level features are reshaped taking into account the dimensions and number of capsules. These capsules apply squashing the vectors, obtaining an instantiation/activity vector the length and orientation of which give the existence of the feature and its instantiation parameters (properties).

large amount of information, exhibiting a greater robustness to variations in position and orientation, scale and illumination than traditional networks.

1) *Routing Between Capsules*: Fig. 1 depicts the standard architecture of a CapsNet. Given an input data  $\mathbf{X} \in \mathbb{R}^{N \times N \times B}$ , where  $N \times N$  denotes the spatial dimensions and  $B$  the number of channels<sup>1</sup>, the model extracts a feature representation from  $\mathbf{X}$  through one or more convolution layers (with data normalization and activation functions but without pooling operators), which process and transform the pixel information into low-level entities, i.e. low-level local features, following Eqs. (1a) and (1b). Indeed, this first part of the CapsNet can be considered as a feature extractor function that maps the raw  $\mathbf{X}$  into a higher dimensional space to prepare the data for capsules.

The obtained feature maps  $\mathbf{X}^l \in \mathbb{R}^{N \times N \times K^l}$  are reshaped into a tensor of  $N \times N \times (K/S \times \mathbb{R}^S)$  entities, where  $S$  is the number of neurons inside each capsule and  $K/S$  the number of capsules at the PrimaryCaps, as Fig. 2 depicts. The output tensor obtained by the  $j$ -th capsule of the  $l$ -th layer is denoted as  $\mathbf{u}_j^l$ , and encodes the spatial information of an feature. These outputs are obtained through a routing system between different capsule layers (except for the first layer) [60]. In particular, it can be obtained as:

$$\mathbf{u}_j^l = \frac{\|\mathbf{s}_j^l\|^2}{1 + \|\mathbf{s}_j^l\|^2} \frac{\mathbf{s}_j^l}{\|\mathbf{s}_j^l\|} \quad (3)$$

where  $\mathbf{s}_j^l$  represents the total input data of the  $j$ -th capsule at the  $l$ -th layer, which is obtained as an agreement function between the lower-level capsules and the current capsule. Indeed, the Eq. (3) performs a non-linear squashing of the input  $\mathbf{s}_j^l$ , forcing it to have a length between 0 and 1 to represent the probability that the feature it represents exists or not within the data, whilst the elements are considered its instantiation parameters, determining the pose and orientation of the feature, as Fig. 2 indicates as ‘‘activity vector’’.

The process to obtain the input vector  $\mathbf{s}_j^l$  is known as routing. Following Eq. (4), the routing-by-agreement mechanism

<sup>1</sup>When processing HSI data, the input to the network are regular patches extracted from the original HSI scene, usually with an odd number of pixels, such that the center pixel will be the target pixel to be classified. Also, to simplify the mathematical notation we consider that the spatial dimensions are maintained along the model.

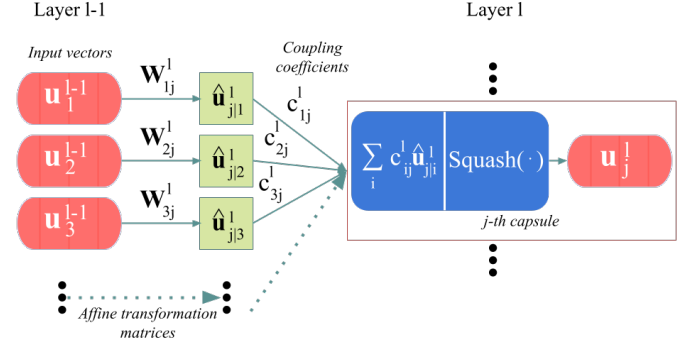


Fig. 3. Details of the dynamic routing and squash function

replaces the pooling function by transformation matrices  $\mathbf{W}_{ij}^l$  which are applied to activity vectors in the previous layer  $\mathbf{u}_i^{l-1}$  to obtain the predictor vectors  $\hat{\mathbf{u}}_{j|i}^l$ . These predictor vectors are the basis for seeking the parents of the  $(l - 1)$ -th layer capsules, i.e. those high-level capsules to which the outputs of the low-level capsules will be routed.

$$\hat{\mathbf{u}}_{j|i}^l = \mathbf{W}_{ij}^l \mathbf{u}_i^{l-1} \quad (4)$$

The parent search is performed by agreement between high and low-level capsules. In this context, the  $j$ -th high-level capsule does not directly receive the predictor vectors as inputs. Instead, its input  $\mathbf{s}_j^l$  is obtained as the weighted sum of all the predictor vectors  $\hat{\mathbf{u}}_{j|i}^l$  and their corresponding coupling coefficients  $c_{ij}^l$  (as Fig. 3 indicates), which are iteratively adjusted by the dynamic routing process following Eq. (5), and are subject to the  $\sum_k c_{ik}^l = 1$  constraint:

$$\mathbf{s}_j^l = \sum_i c_{ij}^l \hat{\mathbf{u}}_{j|i}^l \quad \text{where} \quad c_{ij}^l = \frac{\exp(b_{ij})}{\sum_k \exp(b_{ik})} \quad (5)$$

It can be deduced from Eq. (5) that the larger the scalar product of the prediction vector, the larger the contribution of the corresponding low-level capsule. In this regard, the two capsules will be highly relevant to one another. Parameters  $b_{ik}$  are learned log priors, which represent the probability of the  $i$ -th low-level capsule activating the  $k$ -th high-level capsule, measuring in a certain way the relationship between both capsules. Indeed, this relation evolves as the routing procedure iterates, so each  $b_{ik}$  is initialized to 0 and then it is updated according to the agreement between the prediction made and the obtained output, i.e.  $a_{ik} = \mathbf{u}_k^l \cdot \hat{\mathbf{u}}_{k|i}^l$ , following Eq. (6):

$$\begin{aligned} b_{ik}[t] &= b_{ik}[t-1] + a_{ik}[t-1] \\ &= b_{ik}[t-1] + \left( \mathbf{u}_k^l \cdot \hat{\mathbf{u}}_{k|i}^l \right) [t-1] \\ &= b_{ik}[t-1] + \left( \|\mathbf{u}_k^l\| \|\hat{\mathbf{u}}_{k|i}^l\| \cos(\theta) \right) [t-1], \end{aligned} \quad (6)$$

where  $t$  and  $t - 1$  indicate the current and previous iterations of the routing procedure.

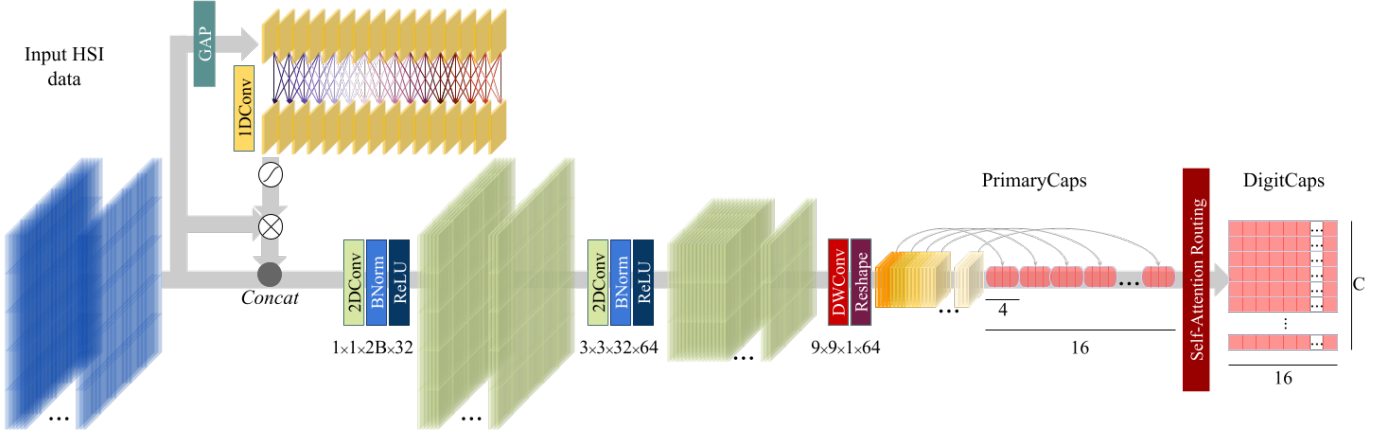


Fig. 4. Graphical overview of the proposed network architecture

2) *CapsNet Optimization*: Focusing on classification task, the parameters of the CapsNet are iteratively adjusted to minimize a loss function. In this sense, the top-level capsule layer  $L$  or DigitCaps contains as many capsules as classes, i.e.  $C$  activity vectors. It is noteworthy that  $\|\mathbf{u}_j^{(L)}\|$  is the probability of belonging to class  $j$ , thus the capsule with the longest instantiation vector (i.e. the highest probability) will determine the class of the original input data  $\mathbf{X}$ . As a result, the loss function is computed following Eq. (7):

$$L_{margin} = \sum_c^C (T_c \max(0, \alpha^+ - \|\mathbf{u}_c^L\|)^2 + \lambda(1 - T_c) \max(0, \|\mathbf{u}_c^L\| - \alpha^-)^2) \quad (7)$$

where  $T_c = 1$  when the data belong to the  $c$ -th class and 0 otherwise. Furthermore,  $\lambda = 0.5$  is a regularization parameter, whilst  $\alpha^+$  and  $\alpha^-$  define two boundaries to force the length of  $\mathbf{u}_c^L$  to lie in the range  $[0.9, 1]$  if  $T_c = 1$  or  $[0, 0.1]$  when  $T_c = 0$ .

The CapsNet can be interpreted as an encoder network. In this sense, a decoder MLP is included to perform data reconstruction. Several fully-connected layers are stacked, and the outputs of the last one are reshaped to obtain the reconstructed data  $\mathbf{X}'$ . As a result the reconstruction loss  $L_{recon} = \|\mathbf{X} - \mathbf{X}'\|$  can be combined with Eq. (7) to enhance the classification performance:  $L_{final} = L_{margin} + L_{recon}$ .

During the backward-step, the gradient signal is calculated for each layer of the network, updating the weights and parameters iteratively. Focusing on capsules, parent capsules update the connections with their child-capsules by modifying the coupling coefficient. However, this process is an intensive and time-consuming task.

### B. Proposed Multiple Attention-Guided CapsNet

Previous works have demonstrated that CapsNets do not need a very large number of instantiation parameters for good performance on simple datasets [67]. In this sense, careful feature selection can improve classification results, whilst attention-based mechanisms can route the desired information

within a network based on the relevance of features. This presents a new way to reduce the computational burden of CapsNets. Indeed, this paper proposes a new CapsNet model, which incorporates different attention-based strategies to improve HSI data classification: i) on the one hand, we include an attention mechanism at the beginning of the convolution-based feature extractor to enhance the initial feature representation, ii) on the other hand, a non-iterative attention-based routing is included to connect a reduced number of capsules efficiently. Fig. 4 provides the graphical representation of the proposed network. Further details are explained below.

1) *Attention-based feature extractor*: As pointed before, the convolution layer produces a set of feature maps by different filters, where each map (channel) contains different local-information. Indeed, only small areas determined by the spectral-spatial size of the convolution kernel will be decisive for the generation of the layer outputs, combining spatial and channel-wise information together. However, this may produce some bias, since the global information is not exploited. On the one hand, some solutions can overcome this problem by extending the receptive field of the model, by means of deeper networks, or by increasing the size of the kernels, but this considerable increases the computational burden of the models whilst producing some negative interference due to internal redundant information. As a result, this strategy is not entirely effective solution and certainly not an efficient one. On the other hand, attention-based strategies can introduce global/contextual information through some statistics used to produce the attention-masks, guiding the models to the most informative features of the data [52], [99], [100]. In this context, the computer vision community has implemented several spatial and spectral attention models [49], [66], [101]–[103], which have been adapted and widely used for HSI data processing [52], [54], [104], [105]. Nevertheless, these models often include more parameters, which results in a higher model complexity and a higher computational burden.

To address these challenges, this paper introduces an efficient channel-based attention module before the first convolution layer of the proposed CapsNet for HSI classification. In this regard, instead of implementing a complete residual

block with several convolutions and the attention mechanism as Wang *et al.* [66], the model directly processes the raw HSI data, capturing local cross-channel interaction. In this sense, two branches have been directly connected to the HSI input: i) the *mask* branch and ii) the *trunk* branch. Within the mask branch, the original patch  $\mathbf{X} \in \mathbb{R}^{N \times N \times B}$  is processed by the channel-wise global average pooling (GAP) described by Eq. (8) [106], which takes the average output of each input channel, whilst condensing the global spatial information of each band into a single scalar without involving learnable parameters.

$$x_t^{GAP} = \text{GAP}_t(\mathbf{X}) = \frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N \mathbf{X}_{ijt}, \forall t \in [1, B] \quad (8)$$

The resulting vector  $\mathbf{x}^{GAP} \in \mathbb{R}^{1 \times 1 \times B}$  is then processed by an 1D convolution layer (1DConv), the kernel of which is adapted according to the number of input channels. As a result, those channels falling within the receptive field of the layer are locally combined, extracting local cross-channel relations. The obtained features are processed by a gate function implemented through the sigmoid ( $\sigma$ ). The resulting vector of descriptors is the mask  $\boldsymbol{\omega} \in \mathbb{R}^{1 \times 1 \times B}$ :

$$\boldsymbol{\omega} = \sigma \left( \text{GAP}(\mathbf{X}) * \mathbf{W}^{1DConv} + \mathbf{b}^{1DConv} \right), \quad (9)$$

where  $\mathbf{W}^{1DConv}$  and  $\mathbf{b}^{1DConv}$  are the kernel parameters and bias of the 1DConv layer, respectively. Moreover, this layer contains zero-padding to maintain the spectral-dimension.

On the other hand, within the trunk branch, the original input data  $\mathbf{X}$  is multiplied by the obtained mask  $\boldsymbol{\omega}$ , following Eq. (2) to obtain the masked input  $\tilde{\mathbf{X}} \in \mathbb{R}^{N \times N \times B}$ . Then, the original input  $\mathbf{X}$  and the masked data  $\tilde{\mathbf{X}}$  are concatenated into  $\tilde{\mathbf{X}} \in \mathbb{R}^{N \times N \times 2B}$  to enrich the input information. This data is further processed by two convolution blocks, which are applied to extract robust spectral-spatial feature. Each block comprises 2DConv-BNorm-ReLU layers, where 2DConv denotes a 2D convolution layer, BNorm is the batch normalization function [107] and ReLU, the Rectified Linear Unit activation function [108], which incorporates Dropout (DO) [109] to avoid the co-adaptation of parameters and enhance the classification performance. The first 2DConv (2DConv-1) preserves the spatial dimensions whilst reducing the number of channels through a kernel size of  $1 \times 1 \times B \times 32$ . Then, the second 2DConv (2DConv-2) extends the number of feature maps while processing the spatial information through a kernel size of  $3 \times 3 \times 32 \times 64$ . The resulting volume with size  $N' \times N' \times 64$  is sent to the PrimaryCaps layer.

2) *Self-attention routing*: A depth-wise convolution layer (DWConv) processes the feature maps obtained by the 2DConv-2, which applies a kernel with size  $N' \times N' \times 1 \times 64$ . Thus, an  $N' \times N'$  filter will be applied separately on each feature map, obtaining a unique coordinate as output. As a result, a vector with  $1 \times 1 \times 64$  elements is obtained. This DWConv significantly reduces the number of trainable parameters that usually are need for the capsules creation. The obtained vector representation is then reshaped to create

the capsules of the PrimaryCaps. In particular, the number of neurons within each capsule has been set to  $S = 4$ , thus there are  $64/4 = 16$  capsules within the PrimaryCaps layer. As a result, the input vector  $1 \times 1 \times 64$  is rearranged into 16 entities of 4 properties in each one. These initial entities are squashed by a modified version of Eq. (10) to obtain the activity vectors  $\mathbf{u}_j^1, \forall j \in [1, 16]$ . In particular, the dubbed squash operation [67] is adopted to avoid vanishing gradient problem as it is more sensitive to small changes:

$$\mathbf{u}_j^1 = \left( 1 - \frac{1}{\exp\|\mathbf{s}_j^1\| + \|\mathbf{s}_j^1\|} \right) \quad (10)$$

where  $\mathbf{s}_j^1$  is the  $j$ -th initial entity obtained by reshaping the output of the DWConv layer. The upper-script <sup>1</sup> indexes the first capsule layer.

These activity vectors are routed to the next layer, DigitCaps, through attention-guided routing. In particular, due to its outstanding ability to capture long-range interactions, self-attention has been adopted between both capsule layers to improve the routing system [67]. Focusing on self-attention, this mechanism considers  $D$  multidimensional inputs which interacts with each other, to produce  $D$  outputs as aggregates of these interactions [96], [110]. Focusing on vectors, for each input  $\mathbf{z}_i \in \mathbb{R}^N$  (with  $i \in [1, D]$ ), *key*  $\mathbf{k}_i \in \mathbb{R}^M$ , *query*  $\mathbf{q}_i \in \mathbb{R}^M$  and *value*  $\mathbf{v}_i \in \mathbb{R}^{M'}$  representations<sup>2</sup> are obtained by multiplying  $\mathbf{z}_i$  with the corresponding set of weights, i.e.,  $\mathbf{k}_i = \mathbf{z}_i \mathbf{W}_i^k$ ,  $\mathbf{q}_i = \mathbf{z}_i \mathbf{W}_i^q$  and  $\mathbf{v}_i = \mathbf{z}_i \mathbf{W}_i^v$  respectively (also, a bias term is introduced in each multiplication). These weights represents affine transformation matrices, i.e.  $\mathbf{W}_i^k, \mathbf{W}_i^q, \mathbf{W}_i^v \in \mathbb{R}^{N \times M}$  that produces different abstractions of the inputs in different subspaces, whilst the meaning of  $\mathbf{k}_i, \mathbf{q}_i$  and  $\mathbf{v}_i$  depends on the application<sup>3</sup>. Moreover, each type of representations can be grouped, thus  $\mathbf{K} \in \mathbb{R}^{D \times M} = [\mathbf{k}_1, \dots, \mathbf{k}_D]^T$ ,  $\mathbf{Q} \in \mathbb{R}^{D \times M} = [\mathbf{q}_1, \dots, \mathbf{q}_D]^T$  and  $\mathbf{V} \in \mathbb{R}^{D \times M'} = [\mathbf{v}_1, \dots, \mathbf{v}_D]^T$ . Then, the self-attention mechanism computes a set of  $D$  score vectors (one per input). There are different score functions, such as the *additive attention* [112], the *multiplicative* or *dot product attention* [113] or the *scale-dot product* [51], among others. For instance, the dot product attention multiplies each input query  $\mathbf{q}_i$  by the keys of all the inputs (including its own), processing the result through a softmax function, i.e.  $\mathbf{s}_i = \text{softmax}(\mathbf{q}_i \mathbf{K})$ . It is noteworthy that each score vector contains  $D$  scores,  $\mathbf{s}_i \in \mathbb{R}^D = [s_{i1}, \dots, s_{iD}]$ , where the score  $s_{ij}$  measure the relevance between the query  $\mathbf{q}_i$  and the key  $\mathbf{k}_j$ . Then, a weighted value representation is calculated for each value  $\mathbf{v}_j$  and its corresponding score  $s_{ij}$ , and the values obtained are summed to produce the output of the  $i$ -th input, as Eq. (11) indicates:

<sup>2</sup>The key, query and value concepts have been inspired by retrieval systems, where, given a query, the system maps it against a set of keys, retrieving those results or values associated with the keys that most closely match the query. It is noteworthy that key and query representations have the same dimension  $M$ , whilst value representations could have a different one  $M'$

<sup>3</sup>For instance, in [111], different convolutions obtains  $\mathbf{k}_i$  and  $\mathbf{q}_i$  as two different automatic-extracted features extracted by several independent convolution layers. These features are combined to create an attention map, which is applied to  $\mathbf{v}_i$  in order to obtain the self-attention feature maps



$$\mathbf{o}_i \in \mathbb{R}^{M'} = \sum_{j=1}^D (s_{ij} \mathbf{v}_j), \forall i \in [1, D] \quad (11)$$

where  $\mathbf{o}_i$  represents the self-attention output features.

Inspired by [67], the self-attention mechanism has been adapted and incorporated to the routing between PrimaryCaps and DigitCaps. Indeed, following Eq. (11), the total input data  $s_j^2$  of the  $j$ -th capsule in the DigitCaps layer is implemented as the weighted sum of all the activity vectors  $\mathbf{u}_i^1$  of the previous layer capsules. In this sense, the connections between the lower and upper-capsules follow a fully-connected scheme.

As pointed in section III-A, during the first step, the PrimaryCaps activity vectors  $\mathbf{u}_i^{l-1}$  are multiplied by a transformation matrix  $\mathbf{W}_{ij}^l$ . In this sense,  $\mathbf{W}^l \in \mathbb{R}^{K/S, C, S, S'}$  can be considered as the tensor that embeds all affine transformation between PrimaryCaps and DigitCaps, containing all matrices  $\mathbf{W}_{ij}^l$ , where dimensions  $K/S = 16$  and  $C$  are the number of capsules in PrimaryCaps and DigitCaps layers (which is the same as the number of land-cover classes), and  $S = 4$  and  $S' = 16$  are the number of neurons embedded within each capsule of the PrimaryCaps and DigitCaps layers, respectively. As a result, Eq. (4) is rewritten as:

$$\hat{\mathbf{U}}^l = (\mathbf{U}^{l-1})^T \times \mathbf{W}^l \quad (12a)$$

$$\hat{\mathbf{U}}_{i,j,:}^l = (\mathbf{u}_{i,:}^{l-1})^T \mathbf{W}_{i,j,:}^l, \forall i \in [1, K/S], j \in [1, C] \quad (12b)$$

where  $\mathbf{U}^{l-1} \in \mathbb{R}^{K/S \times S}$  contains all the PrimaryCaps activity vectors, and  $\hat{\mathbf{U}}^l \in \mathbb{R}^{K/S \times C \times S'}$  all the prediction vectors obtained from each capsule in  $l-1$  for all capsules in  $l$ , i.e., all the prediction vectors for all DigitCaps. Eq. (12b) details the prediction that the bottom-capsule  $i$  makes on the top-capsule  $j$ . Also, Eq. (5) is reformulated in a similar way:

$$\mathbf{S}^l = \hat{\mathbf{U}}^l \times (\mathbf{C}^l + \mathbf{B}) \quad (13a)$$

$$\mathbf{s}_j^l = \hat{\mathbf{U}}_{:,j,:}^l \times (\mathbf{C}_{:,j}^l + \mathbf{B}_{(:,j)}) \quad (13b)$$

where  $\mathbf{S}^l \in \mathbf{C} \times \mathbf{S}'$  contains the total inputs of every single capsule in the  $l$ -th layer. In particular Eq. (13b) provides the formulation to obtain  $\mathbf{s}_j^l \in \mathbb{R}^{S'}$ , i.e. the total input of the  $j$ -th top-capsule. In this context,  $\mathbf{C} \in \mathbb{R}^{K/S, C}$  contains the self-attention coupling coefficients that link the top-layer capsules with the bottom-layer ones, and  $\mathbf{B} \in \mathbb{R}^{K/S, C}$  represents all the log priors. The calculation of the self-attention coefficients  $\mathbf{C}$  is particularly interesting, as they depends on the self-attention tensor  $\mathbf{A} \in \mathbb{R}^{K/S \times K/S \times C}$ , following Eq. (14):

$$\mathbf{C}_{:,j}^l = \frac{\exp\left(\sum_{i=1}^{K/S} \mathbf{A}_{(:,i,j)}\right)}{\sum_{k=1}^C \exp\left(\sum_{t=1}^{K/S} \mathbf{A}_{(:,t,k)}\right)} \quad (14)$$

Indeed,  $\mathbf{C}$  is the softmax of  $\mathbf{A}$ , which contains the score agreement of all possible combinations produced by the  $K/S$  capsules in the bottom-layer. Eqs. (15a) and (15b) provides the formulation. In particular Eq. (15b) shows that  $\mathbf{A}$  is composed of  $C$  symmetric matrices, where the  $\mathbf{A}_{:,j,:}$  matrix collects the attention scores of the  $K/S$  PrimaryCaps for the  $j$ -th

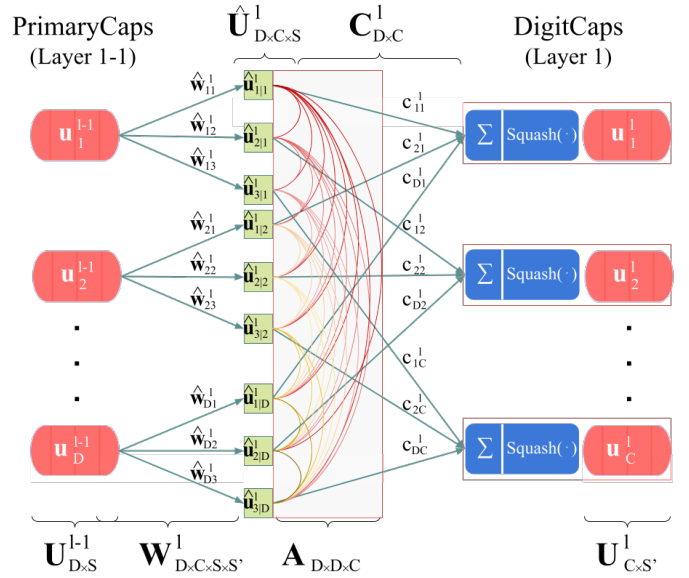


Fig. 5. Self-Attention routing scheme. For simplification  $D$  is set to  $K/S$

DigitCaps output. This is stabilized by the number of neurons that compound the capsules in the PrimaryCaps ( $\sqrt{S}$ ):

$$\mathbf{A} = \frac{\hat{\mathbf{U}}^l \times (\hat{\mathbf{U}}^l)^T}{\sqrt{S}} \quad (15a)$$

$$\mathbf{A}_{:,j,:} = \frac{\hat{\mathbf{U}}_{:,j,:}^l \times (\hat{\mathbf{U}}_{:,j,:}^l)^T}{\sqrt{S}} \quad (15b)$$

Fig. 5 provides a graphical interpretation of the attention-guided routing. Once the total inputs  $\mathbf{S}^l$  are obtained, the dubbed Squash of Eq. (10) is applied to obtain the CapsNet outputs,  $\mathbf{U}^l \in \mathbb{R}^{C \times S'}$ .

3) *Loss and regularization*: As previous HSI-CapsNet [63], the length of the output instantiation vectors is used to determine the land-cover category. The model tries to minimize the loss provided by Eq. (7). Moreover, reconstruction error is introduced as a regularization method to enhance the classification performance.

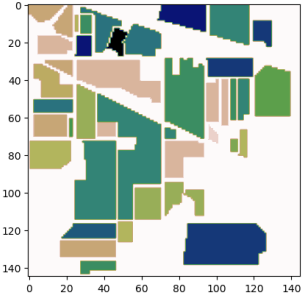
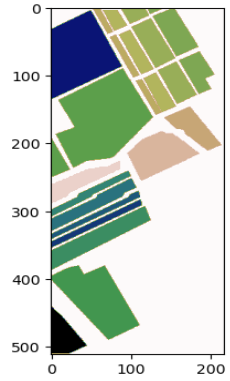
TABLE I  
MULTIPLE ATTENTION-GUIDED CAPSNET ARCHITECTURE

Module	Layer	Dims.	Norm.	Act.	DO	
Attention	GAP	$N \times N$	-	-	-	
	1DConv	Adaptive	-	Sigmoid	-	
F. Extractor	2DConv-1	$1 \times 1 \times B \times 32$	Yes	ReLU	25%	
	2DConv-2	$3 \times 3 \times 32 \times 64$	Yes	ReLU	25%	
Capsules	PrimaryCaps	DWConv	$N' \times N' \times 1 \times 64$	-	-	-
		Reshape Squash	16 Caps. with 4 units	Yes	Squash	-
	DigitCaps	$C$ Caps. with 16 units	Yes	Squash	-	
Recons.	FC1	328	-	ReLU	-	
	FC2	192	-	ReLU	-	
	FC2	$N^2 B$	-	Sigmoid	-	
	Reshape	$N \times N \times B$	-	Sigmoid	-	

Regarding the implementation details, Table I provides the detailed implementation of the proposed network for HSI data classification. It must be noted that input patch size is  $11 \times 11 \times B$ . In this sense, the proposed model takes full advantage of the spectral information contained within the HSI cube. Moreover, the network has been trained with *RAdam* [114]

TABLE II

DESCRIPTION OF THE DATASETS USED FOR THE EVALUATION OF THE PROPOSAL. IMAGE SIZE IS REPRESENTED BY THE HEIGHT AND WIDTH OF EACH DATASET FIGURE. NUMBER OF SAMPLES AMONG THE TYPE OF LAND WITH ITS ASSOCIATED COLOR IS SHOWN IN EACH DATASET TABLE.

INDIAN PINES (IP)			SALINAS VALLEY (SV)			UNIVERSITY OF PAVIA (UP)			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	56975		Background	164624		Background	309157				
	Alfalfa	46		Broccoli-green-weeds-1	2009		Asphalt	6631		Scrub	761				
	Corn-notill	1428		Broccoli-green-weeds-2	3726		Meadows	18649		Willow-swamp	243				
	Corn-min	830		Fallow	1976		Gravel	2099		CP-hammock	256				
	Corn	237		Fallow-rough-plow	1394		Trees	3064		Slash-pine	252				
	Grass/Pasture	483		Fallow-smooth	2678		Painted metal sheets	1345		Oak/Broadleaf	161				
	Grass/Trees	730		Stubble	3959		Bare Soil	5029		Hardwood	229				
	Grass/pasture-mowed	28		Celery	3579		Bitumen	1330		Swap	105				
	Hay-windrowed	478		Grapes-untrained	11271		Self-Blocking Bricks	3682		Graminoid-marsh	431				
	Oats	20		Soil-vinyard-develop	6203		Shadows	947		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			111104	Total samples			207400	Total samples			314368
UNIVERSITY OF HOUSTON (UH)															
						Color	Land cover type	Samples train	Samples test						
							Background	649816							
							Grass-healthy	198	1053						
							Grass-stressed	190	1064						
							Grass-synthetic	192	505						
							Tree	188	1056						
							Soil	186	1056						
							Water	182	143						
							Residential	196	1072						
							Commercial	191	1053						
							Road	193	1059						
							Highway	191	1036						
							Railway	181	1054						
							Parking-lot1	192	1041						
							Parking-lot2	184	285						
							Tennis-court	181	247						
							Running-track	187	473						
						Total samples		2832	12197						

optimizer, considering a batch size of 100 samples, with 200 epochs, and a learning rate of  $1e - 3$

#### IV. EXPERIMENTAL RESULTS

Extensive experimentation has been conducted to evaluate the goodness of the proposed multiple attention-guided CapsNet when dealing with RS-HSI data classification. This section discusses the obtained results.

##### A. Datasets description

Some widely-used HSI scenes have been considered for testing the performance of the proposed model. In particular:

i) *Indian Pines (IP)*, ii) *Salinas Valley (SV)*, iii) *University of Pavia (UP)*, iv) *Kennedy Space Center (KSC)* and v) *University of Houston (UH)* have been studied. Moreover, to prevent over-optimistic and non-realistic results obtained by random sampling techniques, three different disjoint datasets have been selected, in particular *Disjoint University of Pavia (DUP)*, *Disjoint Indian Pine (DIP)* and *Disjoint Houston University (DUH)* [115]. Table II and Figures 6,7 provides the ground-truth details.

In this context, different sensor instruments have collected the data, thus each scene has particular characteristics in terms of spatial resolution and spectral bandwidth, for instance the Airborne Visible Infra-Red Imaging Spectrometer (AVIRIS)

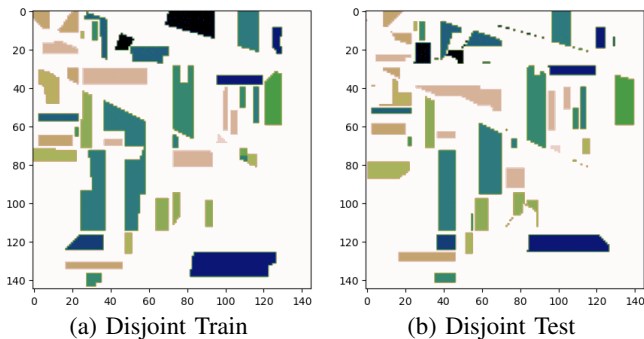


Fig. 6. Dimensional training and test disjoint samples at the left (a) and right (b) for the Indian Pines (DIP) dataset, respectively.

was used to capture IP, SV and KSC datasets, the spectrometer Reflective Optics Spectrographic Imaging System (ROSIS) gathered UP data and the Compact Airborne Spectrographic Imager (CASI) collected the UH data. In terms of image size, the original IP dataset is composed of  $145 \times 145$  pixels with 224 spectral bands which are usually reduced to 200, KSC has 224 images of  $521 \times 614$ , UP comprises  $610 \times 340$  pixels with 103 channels, the  $512 \times 217$  pixels of SV contains 204 bands, and UH has  $349 \times 1905$  pixels with 144 channels.

Furthermore, these datasets were captured over different zones, from an agricultural zone in Indiana (USA), the Kennedy Space Center from Florida, the city of Pavia in Italy, a rural area with crop fields in California (USA) and the University of Houston and its surroundings, respectively. In terms of different pixel information, IP is composed of 16 different classes, which mainly are crop fields, KSC has 13 different classes which only represent a low portion of the image since most pixels contain background information. UP, UH and SV have a similar problem, although they are composed of 9 classes typical of an urban environment (except SV). Regarding the wavelength and the spatial resolution, IP was captured over the range of  $400 - 2500nm$  with a spatial resolution of  $20m$  per pixel, KSC also ranges from  $400 - 2500nm$  and  $18m$ , UP is within the range of  $430 - 860nm$  and  $1.3m$ , SV in the  $400 - 2450nm$  and  $3.7m$ , and UH captures  $380 - 1050nm$  with  $2.5m$ , respectively.

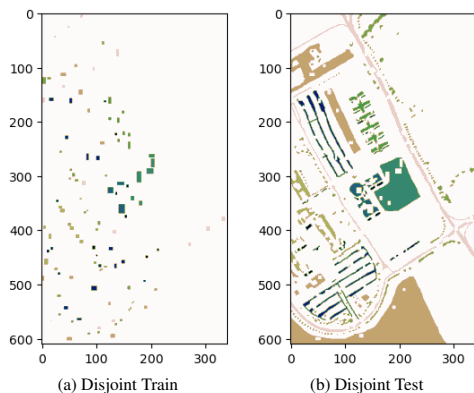


Fig. 7. Dimensional training and test disjoint samples at the left (a) and right (b) for the University of Pavia (DUP) dataset, respectively.

Focusing on disjoint datasets, DUP, DIP and DUH scenes

are used to overcome the limitation of using the same scene for both training and testing. Thus, exclusive samples are used for train and test steps without spatial overlapping between them. This adds a challenge for the classification, since the class balance is randomized, i.e. the possibility that a type of material or plant contained in a specific split of the scene could not be included is high. This also helps to prove the adaptability and robustness of the classification model. These type of datasets have been proved to be very efficient for HSIs classification since the application of the kernel from CNNs could overlap information between train and test data [26].

## B. Environment description

To conduct the experiments, the newest software and hardware resources have been employed. As hardware resources, the Intel i9-9940X processor with 128GB of DDR4 RAM is exploited. Regarding the graphic processing unit (GPU), the NVIDIA Titan RTX with 24GB of DDR4 RAM is used. Finally, Keras library has been considered for code implementation, taking advantage of its facilities and functionalities, with TensorFlow as the back-end.

## C. Results and discussion

In order to correctly evaluate the classification results, different metric have been used: i) the *Overall Accuracy* (OA), ii) the *Average Accuracy* (AA) and iii) the *Kappa metric* (K). These metrics are used as the three most quantitative methods in the literature to correctly determine the classification quality of different classification models. In addition, we have compared the proposed capsule-based network with current state-of-art methods. These methods are: i) *Random Forest* (RF), ii) *Multiple Logistic Regression* (MLR), iii) *Support Vector Machine* (SVM), iv) *Multilayer Perceptron* (MLP), v) *Recurrent Neural Network* (RNN), vi) *Long Short-Term Memory* network (LSTM), vii) *Gated Recurrent Units* (GRU), viii) *2DCNN* and ix) *3DCNN*, x) *Deep Pyramidial Residual Network* (pResNet) [44], xi) *3D Generative Adversarial Minority Oversampling* (3D-GAMO) [38], xii) *Graph Convolutional Networks for Hyperspectral Image Classification* (GCN-Fus) [39] and xiii) *CapsNet* [63]. It should be noted that these methods have been implemented with the recommended parameters, whilst spectral-spatial methods have been adjusted to work with  $11 \times 11$  input patch size to realize a fair comparison with our proposal.

The experiments have been organized into three different groups:

- The first experiment performs an ablation study to analyze the impact of those improvements introduced on the original capsule model. In this sense, and considering the 5% of labelled data of KSC scene, the standard CapsNet model proposed by [63] has been compared with a *dynamic-routing*-based CapsNet restricted to have the same number of capsules (and the same number of properties per capsule) as the proposed model, and with the different variations of the implemented model. Obtained results in terms of OA, AA, Kappa coefficient, runtime and number of parameters are shown in Table III.

TABLE III  
ABLATION STUDY OVER KSC SCENE USING THE 5% OF THE DATA.

Metric	CapsNet [63]	Eq-CapsNet	S-Prop	FE+S-Prop	C+S-Prop	C+con+S-Prop	Proposed
OA(%)	95.46±0.49	94.57±0.69	96.65±0.64	97.08±0.82	96.69±0.89	96.85±0.86	<b>97.73</b> ±0.51
AA(%)	91.83±0.87	89.32±1.86	93.43±1.63	94.33±1.73	93.32±2.08	93.75±1.81	<b>95.71</b> ±1.20
K(x100)	94.95±0.54	93.95±0.77	96.27±0.71	96.75±0.91	96.32±1.00	96.49±0.96	<b>97.47</b> ±0.57
Runtime(s)	<b>156.4</b> ±0.55	175.97±1.39	172.77±1.02	177.66±2.78	176.13±2.26	175.84±1.47	179.71±1.38
Time/Epoch(s)	1.564±0.006	0.880±0.007	<b>0.864</b> ±0.005	0.888±0.014	0.881±0.011	0.874±0.007	0.899±0.007
Parameters	7847352	4707768	4381176	<b>4304024</b>	4381176	4482556	4309660

- The second experiment compares different care mechanisms that can be implemented at the beginning of the network. In particular, the proposed model with efficient channel attention has been compared with four different models, i.e. the baseline with dynamic-routing and no attention mechanism, an implementation with self-attention routing but without attention mechanism, an implementation with self-attention routing and spatial attention, and an implementation with self-attention routing and convolutional block attention module (CBAM) [102]. Obtained results are shown in Table IV considering the 5% of labelled data of KSC scene.
- The behaviour of the proposed network has been evaluated over IP, SV, UP and KSC scenes considering different number of training samples, which are randomly extracted from the HSI scenes. In particular, the models have been trained with 1%, 3% and 5% of SV and UP, and with 3%, 5% and 10% of KSC and IP because of their high spectral mixture. Fig. 8 provides the obtained results in terms of OA and standard deviation.
- The second experiment evaluates the generalization ability of the proposed network when non-overlapped data is introduced during training and testing stages. In particular, the fixed training and test samples of the disjoint datasets DIP, DUP and DUH, have been considered to perform a realistic evaluation, avoiding the optimistic results obtained by random sample selection methods. Tables V, VI and VII provide the obtained results in terms of OA, AA and Kappa coefficient. Also, Figs. 9, 10 and 11 depict the classification maps.
- In the third experiment different input patch-sizes have been considered to evaluate the impact of the spatial information within the spectral-spatial classifiers. In particular, we have considered  $5 \times 5$ ,  $7 \times 7$ ,  $9 \times 9$  and  $11 \times 11$  input pixels for the IP and UP scenes. Table VIII shows the obtained results.

1) *Performance analysis through ablation study:* This experiment evaluates the impact of the improvements that have been included in the proposed capsule-based architecture for HSI data classification. As described above, the proposed network incorporates an attention mechanism at the beginning of the processing stack, two feature extraction (FE) groups composed of convolution-normalisation-activation layers, and the self-attention guided routing algorithm (comprising 16 capsules of 4 properties in the PrimaryCaps layer). In order to evaluate the impact of each part in terms of classification accuracy (OA, AA and Kappa) and computational performance

(runtime and number of parameters), several implementations have been considered: i) **CapsNet** is the standard capsule-based model of [63] without attention mechanisms and with dynamic-routing (it should be noted that it was implemented in pytorch, whilst the rest of the models are in keras); ii) **Eq-CapsNet**, a standard network equivalent to the proposed model, without attention mechanism but with two FE-groups and with dynamic-routing, considering 16 capsules of 4 properties in the PrimaryCaps layer; iii) **S-Prop**, a capsule-model with only self-attention guided routing, considering 16 capsules of 4 properties in the PrimaryCaps layer; iv) **FE+S-Prop**, a capsule-model with two FE-groups and the self-attention guided routing; v) **C+S-Prop**, a capsule-model with self-attention guided routing and the efficient channel attention mechanism at the bottom of the model, where the masked inputs  $\tilde{X}$  are forward-propagated; vi) **C+con+S-Prop** a capsule-model with self-attention guided routing and the efficient channel attention mechanism at the bottom of the model, where the concatenation of original inputs  $X$  and masked inputs  $\tilde{X}$  are forward-propagated as  $\tilde{X}$ , and finally vii) **Proposed**, the proposed capsule-model with efficient channel attention, where original inputs and masked inputs are concatenated into  $\tilde{X}$  and processed by two FE-groups, to extract refined features that will compose the capsules with self-attention routing.

All seven models have been trained and evaluated on the challenging KSC scene due to their high spectral complexity and the sparse spatial information, using 5% of labelled samples. In particular, the *CapsNet* model is trained with 100 epochs, whilst the rest of the models are trained with 200 epochs. Table III provides the obtained results. Comparing the models based on dynamic-routing, i.e. *CapsNet* and *Eq-CapsNet*, it can be seen that the former has many more parameters than the latter (3 139 584 parameters more, which is to be expected, as it has many more and larger capsules). Nonetheless, its computation time is shorter (it is 19.57 seconds faster), as it was implemented in PyTorch with 100 training epochs, while the second one was implemented with Keras with 200 training epochs. In this sense, comparing the runtime exhibited by the *CapsNet* model is not fair due to the large differences between the frameworks. On the other hand, the accuracy achieved by the *CapsNet* is better than that obtained by the *Eq-CapsNet*.

Comparing the *Eq-CapsNet* with the *S-Prop* model, an improvement in accuracy can be observed when the dynamic-routing is replaced by the self-attention routing. Furthermore, the number of parameters is reduced by 326 592 parameters

and the runtime is slightly improved (*S-Prop* is 3.2 seconds faster than *Eq-CapsNet*). This is explained by the fact that the self-attention mechanism is more time consuming, although it can be optimized through the implementation of parallel optimisations. By including the two FE-groups at the beginning of the model in *FE+S-Prop*, a more refined treatment of the data is performed, resulting in a significantly higher OA in comparison with *Eq-CapsNet* and *S-Prop*. It is worth noting that despite deepening the network, the processing performed by the  $1 \times 1 \times B \times 32$  and  $3 \times 3 \times 32 \times 64$  convolutions is very efficient, slightly increasing the runtime (it is 1.69 and 4.89 seconds slower than *Eq-CapsNet* and *S-Prop*, respectively). Indeed, they are extracting useful spectral-spatial information whilst reducing the input volume dimensions. Thus, the creation of the PrimaryCaps layer is more efficient, reducing the number of parameters with respect to the baseline *Eq-CapsNet* and *S-Prop* by 403 744 and 77 152 fewer parameters respectively.

In contrast, when comparing *S-Prop* and *C+S-Prop*, although *C+S-Prop* certainly produces a slightly better accuracy result, it does not make as big a difference as the *FE+S-Prop* model, whilst the runtime is increased in 3.36 seconds. In fact the differences in the runtime are hardly noticeable, considering that the times are obtained every 200 epochs, i.e., the *S-Prop* wastes 0.86 seconds per epoch and *C+S-Prop*, 0.88 seconds. Rather than using only the masked samples, if the original  $\mathbf{X}$  and masked samples  $\tilde{\mathbf{X}}$  are concatenated into  $\tilde{\mathbf{X}}$  as in *C+con+S-Prop*, another slight improvement in classification is obtained in comparison with *C+S-Prop*, in exchange for increasing the number of parameters. On the one hand, this increases the number of parameters in the model (which may worsen the overfitting problem), but on the other hand it provides more descriptive data to the network (which improves the final classification). Finally, by including the two FE-groups (thus obtaining the *Proposed* model), the number of parameters is reduced and the data is refined. As a result, the *Proposed* model achieves the best classification accuracy, exhibiting a smaller number of parameters compared to the baseline model, *Eq-CapsNet*.

A last comparison can be conducted between the *CapsNet* and the *Proposed* models, regarding the efficiency of the models. In this sense, previous comparisons regarding the runtime achieved by *CapsNet* and the rest of the models were not fair due to the large difference in the programming framework (pytorch vs. keras) and the number of epochs used (100 vs. 200). In this sense, although programming frameworks are different, the time per epoch measured in seconds (Time/Epoch) has been obtained for each considered model. As it is observed, the *CapsNet* is the slowest model due to the large number of parameters to be trained. In this sense, the *Proposed* model is almost two times faster than the *CapsNet*, therefore it should process two times more images during the inference in the same time. On the other hand, the baseline *Eq-CapsNet* model is significantly faster than the *CapsNet* model, also due to the decrease in the number of parameters, and slightly faster than the proposed model, despite having more parameters than the *Proposed* model. Logically, the two FE-stages introduce more computational load, as it is observed

in the *FE+S-Prop* and *Proposed* models. Indeed, focusing on *S-Prop*, *C+S-Prop* and *C+con+S-Prop* models, it could be observed that self-attention routing is quite efficient, exhibiting the lowest time per epoch, whilst the channel attention mechanism in *C+S-Prop* increases the time, as masked features do not provide all of the desired information to enhance the learning process. On the contrary, the concatenation of original and masked features in *C+con+S-Prop* increases the time (since more data must be processed and therefore the number of parameters increases) but improves the learning process. In this sense, adding the two stages of FE significantly improves the learning process, i.e., the classification accuracy by slightly increasing the complexity of the model, as we can observe by comparing the *C+con+S-Prop* and the *Proposed* models. We can conclude that the *Proposed* model is more efficient and effective than the previous ones, providing a good trade-off between runtime, time per epoch, number of parameters and accuracy achieved.

2) *Comparison between different attention mechanism*: To complete the evaluation of the proposed method, the second experiment conducts a study between different attention mechanisms. In this sense, different implementations have been implemented: i) **Eq-CapsNet** is the standard network equivalent to the proposed model, with dynamic-routing; ii) **S-Prop2** is the capsule-model with self-attention routing and no attention mechanism; iii) **Sp+S-Prop** is implemented as a capsule-model with self-attention routing and spatial attention technique at the beginning of the network [102]; iv) **CBAM+S-Prop** is implemented as a capsule-model with self-attention routing and CBAM at the beginning of the network [102], and v) the **Proposed** model with self-attention and efficient channel attention [66]. For all attention mechanisms, the masked features have been concatenated to the original features and sent to the FE-groups.

TABLE IV  
EVALUATION OF DIFFERENT ATTENTION MECHANISM OVER KSC (5%)

Metric	Eq-CapsNet	S-Prop2	Sp+S-Prop	CBAM+S-Prop	Proposed
OA	94.57±0.69	97.08±0.82	97.63±0.71	96.94±0.52	<b>97.73</b> ±0.51
AA	89.32±1.86	94.33±1.73	95.38±1.60	94.57±0.77	<b>95.71</b> ±1.20
K(x100)	93.95±0.77	96.75±0.91	97.36±0.80	96.59±0.58	<b>97.47</b> ±0.57
Runtime	<b>175.97</b> ±1.39	177.66±2.78	182.53±1.93	200.18±2.61	179.71±1.38
Parameters	4707768	<b>4304024</b>	4381274	4389216	4309660

These models have been trained and evaluated on the challenging KSC scene, considering 5% of training data. Table IV provides the obtained results in terms of OA, AA, Kappa coefficient, runtime and number of parameters. Obtained results demonstrate that the *Proposed* network reaches the best accuracy results. Focusing on runtime and parameters, the baseline *Eq-CapsNet* is the model with the most parameters, precisely because dynamic-routing generates a large number of instantiation parameters. Nevertheless, its processing is slightly faster, but its accuracy results are by far the worst. On the other hand, the *S-Prop2* comprises the lowest number of parameters, improving accuracy results compared to the baseline. Focusing on attention-based *Sp+S-Prop* and *CBAM+S-Prop*, they introduce a large number of parameters, increasing the complexity of the model and the runtime. Indeed, focusing on *CBAM+S-Prop*, the CBAM mechanism introduces

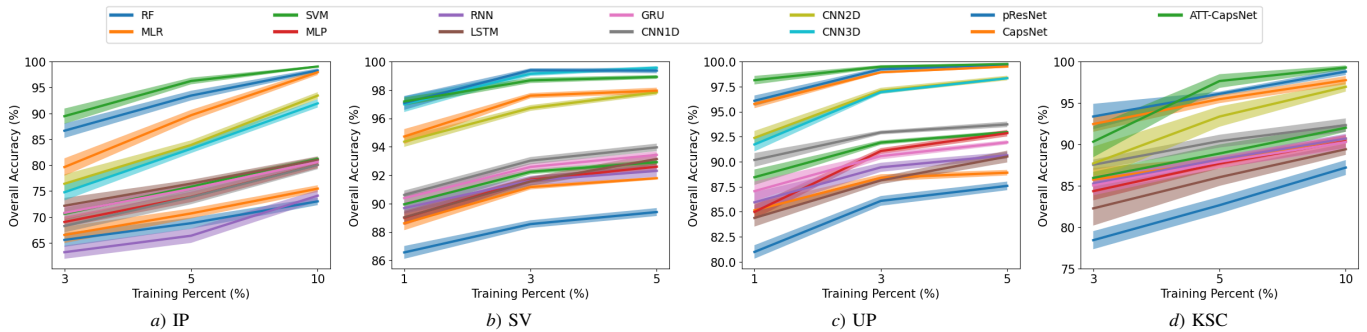


Fig. 8. Overall Accuracy (OA) using different amount of training percentages for the IP, SV, UP and KSC datasets. This is obtained using random sampling.

TABLE V

ACCURACY FOR THE CLASSIFICATION OF THE DIFFERENT CLASSES AMONG WITH THE GLOBAL OA, AA AND K(x100) METRICS FOR THE DIFFERENT STUDIED METHODS. RESULTS ARE CALCULATED FOR THE DIP SCENE.

Class	RF	MLR	SVM	MLP	RNN	LSTM	GRU	CNN1D	CNN2D	CNN3D	pResNet	3D-GAMO	CapsNet	ATT-CapsNet
0	34.4±3.67	68.0±0.0	<b>96.0±0.0</b>	69.2±9.13	68.8±15.57	84.4±16.04	93.6±5.43	72.0±14.31	66.82±18.07	39.09±12.4	78.0±12.43	70.0	88.8±5.95	94.4±5.99
1	51.42±0.38	78.07±0.0	80.74±0.0	84.21±2.12	78.33±2.87	80.4±3.57	79.9±2.48	80.67±3.53	69.8±5.0	68.19±2.98	87.24±3.84	33.33	90.4±1.06	<b>93.07</b> ±2.37
2	44.6±0.69	59.41±0.0	70.79±0.0	70.25±4.79	60.0±7.67	65.52±4.61	64.7±5.51	71.58±6.67	57.25±6.49	80.19±4.59	71.73±6.1	<b>97.33</b>	57.06±3.65	88.44±7.11
3	27.17±2.32	25.25±0.0	51.52±0.0	43.33±8.83	33.43±6.17	45.86±6.26	49.29±9.45	49.6±16.62	26.18±2.89	39.21±17.02	43.33±1.99	<b>74.23</b>	42.82±2.83	54.85±9.37
4	80.47±1.04	88.32±0.0	87.59±0.0	86.2±0.93	84.89±4.09	87.52±1.04	87.55±2.05	88.25±0.76	62.59±7.43	85.63±2.18	76.86±11.62	<b>96.67</b>	83.03±5.06	91.46±0.75
5	95.68±0.38	96.89±0.0	96.61±0.0	97.29±0.71	93.73±0.94	97.26±1.0	97.26±0.52	97.71±0.57	97.81±1.21	<b>99.15</b> ±0.61	90.57±3.01	37.71	96.36±1.45	98.53±0.88
6	30.0±24.49	50.0±0.0	<b>100.0</b> ±0.0	70.0±45.83	0.0±0.0	10.0±20.0	0.0±0.0	5.0±15.0	0.0±0.0	0.0±0.0	0.0±0.0	76.16	0.0±0.0	0.0±0.0
7	<b>100.0</b> ±0.0	99.2±0.0	98.8±0.0	97.8±1.05	98.88±1.09	99.52±0.5	98.56±2.09	99.72±0.47	94.0±5.93	95.11±4.93	94.6±2.74	98.53	99.56±0.52	100.0±0.0
8	9.0±5.39	40.0±0.0	70.0±0.0	68.0±17.2	53.0±10.05	60.0±11.83	63.0±10.05	<b>80.0</b> ±12.65	75.56±11.97	44.44±19.88	67.0±14.87	62.29	69.0±5.39	69.0±14.46
9	8.27±0.39	56.14±0.1	81.91±0.0	79.11±2.5	74.51±6.25	75.13±4.95	83.9±4.33	78.27±3.44	80.64±2.06	57.0±6.16	83.94±3.67	86.29	90.28±1.84	<b>93.02</b> ±2.69
10	89.58±0.48	81.64±0.05	87.51±0.0	82.89±2.94	79.05±2.64	80.82±2.9	81.43±1.88	85.0±3.53	81.49±1.08	83.32±0.88	86.7±1.7	<b>96.52</b>	86.86±2.46	92.52±2.83
11	26.67±1.11	68.44±0.0	80.5±0.0	77.45±3.4	66.42±9.82	77.91±4.2	78.58±3.91	82.34±3.16	53.19±4.87	63.03±7.39	59.97±4.4	61.63	71.81±5.28	<b>86.74</b> ±4.37
12	88.38±0.57	96.25±0.0	93.75±0.0	97.0±1.15	96.12±1.89	97.38±1.04	96.62±0.57	97.88±1.12	<b>99.31</b> ±0.93	95.0±2.58	91.25±2.96	89.13	96.12±2.98	97.5±1.85
13	92.09±0.21	89.98±0.09	91.93±0.0	93.94±1.7	91.69±2.65	94.18±2.09	94.35±2.71	92.7±1.22	94.8±2.09	94.61±1.93	98.4±1.18	98.23	97.36±0.92	<b>99.67</b> ±0.59
14	36.57±1.41	82.83±0.0	78.79±0.0	86.26±3.81	82.32±0.52	83.64±4.44	82.83±2.96	81.52±3.03	87.75±10.93	28.2±14.56	63.03±9.32	<b>89.48</b>	81.92±4.73	69.5±12.57
15	93.41±0.68	93.18±0.0	88.64±0.0	85.68±4.07	88.86±2.95	90.91±1.76	93.41±5.97	90.23±6.1	90.25±7.94	<b>95.5</b> ±3.67	90.07±0.06	91.92	87.27±5.94	89.77±4.57
OA(%)	65.68±0.14	78.16±0.02	85.08±0.0	84.01±0.29	79.38±0.53	82.47±0.34	83.53±0.5	84.5±0.56	77.51±1.13	77.99±0.91	83.66±1.02	86.96	85.71±0.54	<b>92.27</b> ±0.86
AA(%)	56.73±1.7	73.35±0.01	<b>84.69</b> ±0.0	80.54±2.97	71.88±1.25	76.9±1.25	77.81±0.85	78.28±1.78	71.09±1.31	66.73±1.93	73.91±1.08	78.72	77.29±0.72	82.4±1.09
K(x100)	59.86±0.17	75.0±0.02	82.98±0.0	81.79±0.34	76.47±0.64	80.03±0.4	81.25±0.59	82.33±0.64	74.31±1.29	74.79±1.08	81.4±1.16	85.17	83.69±0.62	<b>91.21</b> ±0.98

TABLE VI

ACCURACY FOR THE CLASSIFICATION OF THE DIFFERENT CLASSES AMONG WITH THE GLOBAL OA, AA AND K(x100) METRICS FOR THE DIFFERENT STUDIED METHODS. RESULTS ARE CALCULATED FOR THE DUP SCENE.

Class	RF	MLR	SVM	MLP	RNN	LSTM	GRU	CNN1D	CNN2D	CNN3D	pResNet	3D-GAMO	GCN-Fus	CapsNet	ATT-CapsNet
0	79.74±0.22	77.7±0.01	82.23±0.0	85.41±1.28	79.65±3.46	82.36±3.85	80.79±3.96	88.63±2.13	82.81±1.78	89.72±2.29	87.92±3.53	<b>97.4</b>	96.67	93.04±1.96	95.39±1.31
1	55.27±0.36	58.78±0.01	65.81±0.0	74.11±2.22	68.66±4.31	78.1±2.54	78.08±5.71	89.59±3.17	94.56±0.93	76.1±2.93	97.72±0.59	<b>99.49</b>	97.6	98.81±0.45	97.55±0.99
2	45.48±0.41	67.22±0.0	66.72±0.0	70.82±4.47	63.45±7.83	63.11±4.64	65.41±8.11	73.94±6.65	65.14±2.82	80.61±5.37	71.99±5.45	<b>91.71</b>	84.49	56.13±9.21	74.49±9.1
3	98.73±0.07	74.27±0.06	97.77±0.0	91.64±2.64	92.11±2.73	96.72±7.1	94.62±1.15	92.46±3.85	97.0±1.29	78.72±3.64	96.58±2.97	71.31	89.95	<b>98.75</b> ±0.46	97.49±1.18
4	99.08±0.1	98.89±0.04	99.37±0.0	99.36±0.12	99.2±0.29	99.09±0.36	99.43±0.14	99.39±0.85	98.38±1.7	99.62±0.52	98.82±0.71	98.3	<b>99.64</b>	99.46±0.25	99.17±0.32
5	78.92±0.45	93.53±0.01	91.62±0.0	91.01±1.9	86.41±4.24	67.91±6.0	75.99±15.18	86.18±5.03	65.03±4.19	96.73±1.91	70.62±7.68	<b>97.99</b>	90.56	92.82±4.89	93.26±4.46
6	79.42±1.1	85.1±0.04	87.36±0.0	88.52±2.04	86.81±10.55	86.37±4.87	87.57±4.16	89.41±2.3	75.91±4.76	<b>97.42</b> ±1.34	84.93±4.13	97.25	78.27	87.14±1.21	94.69±5.25
7	90.74±0.24	87.57±0.01	90.46±0.0	89.81±2.04	87.06±6.01	88.85±2.86	88.2±5.69	89.29±2.83	96.47±0.92	86.14±5.97	96.38±1.3	94.1	71.73	<b>98.26</b> ±0.81	97.74±0.17
8	97.58±0.17	<b>99.23</b> ±0.04	93.71±0.0	98.26±1.07	94.82±4.44	98.08±1.96	95.31±6.07	95.91±6.19	92.22±1.53	97.94±0.76	97.07±1.28	92.09	98.04	96.83±1.92	99.14±0.55
OA(%)	70.18±0.14	72.23±0.01	77.8±0.0	81.81±0.61	77.25±1.11	80.37±0.39	80.92±1.02	88.92±0.99	87.94±0.41	83.46±1.16	91.42±0.94	93.9	92.2	94.92±0.47	<b>95.69</b> ±0.85
AA(%)	80.55±0.14	82.48±0.01	86.12±0.0	87.66±0.6	84.24±1.11	84.51±0.7	85.04±1.69	89.42±0.97	85.28±0.77	89.22±0.77	89.12±1.11	93.29	89.66	91.25±0.95	<b>94.32</b> ±1.44
K(x100)	63.04±0.14	65.44±0.01	72.06±0.0	76.63±0.69	71.08±1.15	74.36±0.42	75.17±1.27	85.25±1.19	83.63±0.57	78.68±1.38	88.29±1.32	91.86	89.51	93.13±0.65	<b>94.19</b> ±1.14

TABLE VII

ACCURACY FOR THE CLASSIFICATION OF THE DIFFERENT CLASSES AMONG WITH THE GLOBAL OA, AA AND K(x100) METRICS FOR THE DIFFERENT STUDIED METHODS. RESULTS ARE CALCULATED FOR THE UH SCENE.

Class	RF	MLR	SVM	MLP	RNN	LSTM	GRU	CNN1D	CNN2D	CNN3D	pResNet	GCN-Fus	CapsNet	ATT-CapsNet
0	82.49±0.11	82.21±0.04	82.34±0.0	81.38±0.41	81.86±0.56	82.6±0.35	82.49±0.47	82.48±0.81	80.83±1.05	82.41±0.76	81.83±0.52	<b>83.86</b>	82.8±0.22	82.7±0.24
1	83.37±0.09	82.46±0.05	83.36±0.0	82.25±1.28	82.5±1.03	80.19±1.08	81.48±1.84	89.06±5.88	88.49±4.91	92.69±4.73	84.83±0.58	<b>98.59</b>	85.34±0.63	87.76±4.3
2	97.88±0.25	<b>99.8</b> ±0.0	99.8±0.0	99.66±0.09	99.7±0.16	99.76±0.12	99.8±0.15	99.74±0.24	94.63±2.3	98.7±0.71	95.47±2.01	83.37	98.97±0.66	99.07±0.69
3	91.58±0.18	98.3±0.0	98.96±0.0	87.34±0.46	95.67±2.45	91.72±1.03	94.87±3.35	93.44±4.36	88.38±3.49	86.07±1.29	86.91±3.49	<b>98.96</b>	89.6±0.92	92.12±2.39
4	96.69±0.15	97.44±0.0	98.77±0.0	97.4±0.35	97.75±0.32	97.54±0.51	97.08±0.6	98.88±0.4	99.98±0.04	<b>100.0</b> ±0.0	99.92±0.23	99.72	100.0±0.0	100.0±0.0
5	<b>98.95</b> ±0.35	94.41±0.0	97.9±0.0	94.76±0.64	95.31±0.32	96.5±1.85	98.67±1.27	95.45±1.3	95.5±1.76	98.14±2.0	95.66±3.45	96.5	96.01±0.32	96.36±1.28
6	75.46±0.28	73.41±0.0	77.43±0.0	76.26±2.56	82.02±1.9	79.99±1.81	78.97±1.66	81.99±5.62	77.87±3.98	78.47±1.56	79.25±4.86	<b>89.55</b>	80.26±2.18	78.41±2.78
7	32.95±0.27	63.82±0.0	60.3±0.0	59.86±7.25	40.91±0.82	39.66±1.47	51.41±6.08	80.35±6.16	74.87±2.07	79.17±3.47	80.46±4.7	<b>89.36</b>	78.65±2.43	78.03±2.29
8	69.67±0.49	70.25±0.03	76.77±0.0	71.92±2.52	77.36±1.53	77.0±2.55	77.3±2.57	77.24±4.27	81.39±2.63	84.04±2.67	81.15±1.47	83.29	<b>90.24</b> ±1.96	89.2±2.84
9	43.47±0.51	55.6±0.0	61.29±0.0	50.12±3.9	46.81±1.92	49.7±3.1	51.25±5.27	74.29±13.41	71.19±12.67	62.93±7.18	<b>80.12</b> ±14.92	79.25	70.09±6.17	72.14±9.54
10	69.98±0.2	74.19±0.0	80.55±0.0	76.46±0.98	75.69±1.12	77.24±1.27	79.06±0.96	82.03±4.3	93.8±2.5	97.15±1.05	94.65±4.68	79.89	97.2±0.74	<b>98.55</b> ±1.03
11	54.01±0.91	70.43±0.04	79.92±0.0	78.51±4.78	76.58±3.5	82.33±3.81	82.84±3.59	90.12±3.82	96.52±1.89	<b>98.84</b> ±0.78	97.59±1.19	79.15	98.53±0.54	98.25±1.81
12	60.46±0.77	67.72±0.0	70.88±0.0	72.88±2.55	69.68±2.24	70.14±3.08	71.65±3.05	76.67±4.09	81.25±0.29	81.76±2.6	80.49±3.78	87.72	86.66±1.47	<b>89.16</b> ±0.66
13	98.91±0.26	98.79±0.0	<b>100.0</b> ±0.0	99.6±0.31	99.96±0.12	99.19±0.48	99.72±0.41	99.51±0.3	99.91±0.27	99.91±0.27	99.88±3.36	93.93	100.0±0.0	100.0±0.0
14	97.51±0.25	95.56±0.0	96.41±0.0	97.89±0.48	97.95±0.44	98.18±0.33	98.39±0.48	98.18±0.42	99.06±0.96	<b>100.0</b> ±0.0	92.77±4.71	98.94	99.96±0.13	99.94±0.1
OA(%)	72.98±0.14	78.97±0.01	81.86±0.0	78.63±0.44	78.22±0.36	78.29±0.31	79.99±0.61	86.34±0.65	86.57±1.03	87.56±0.68	87.49±1.2	88.62	88.6±0.54	<b>89.05</b> ±0.78
AA(%)	76.89±0.15	81.63±0.01	84											

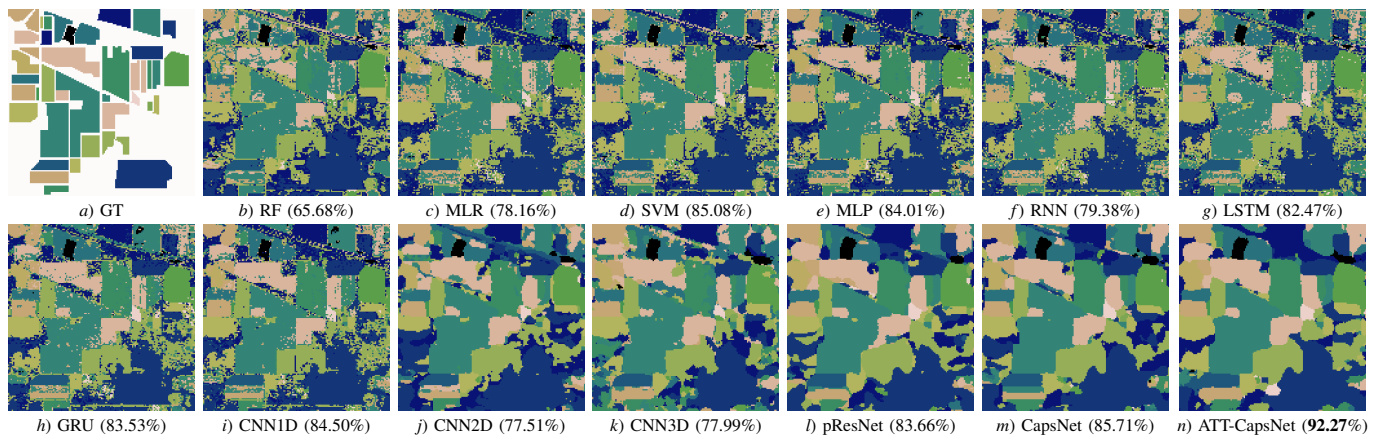


Fig. 9. Visualization of the Ground truth (a) map and the respective classification map for the previous noted methods for the DIP dataset. Accuracy is also included for every method.

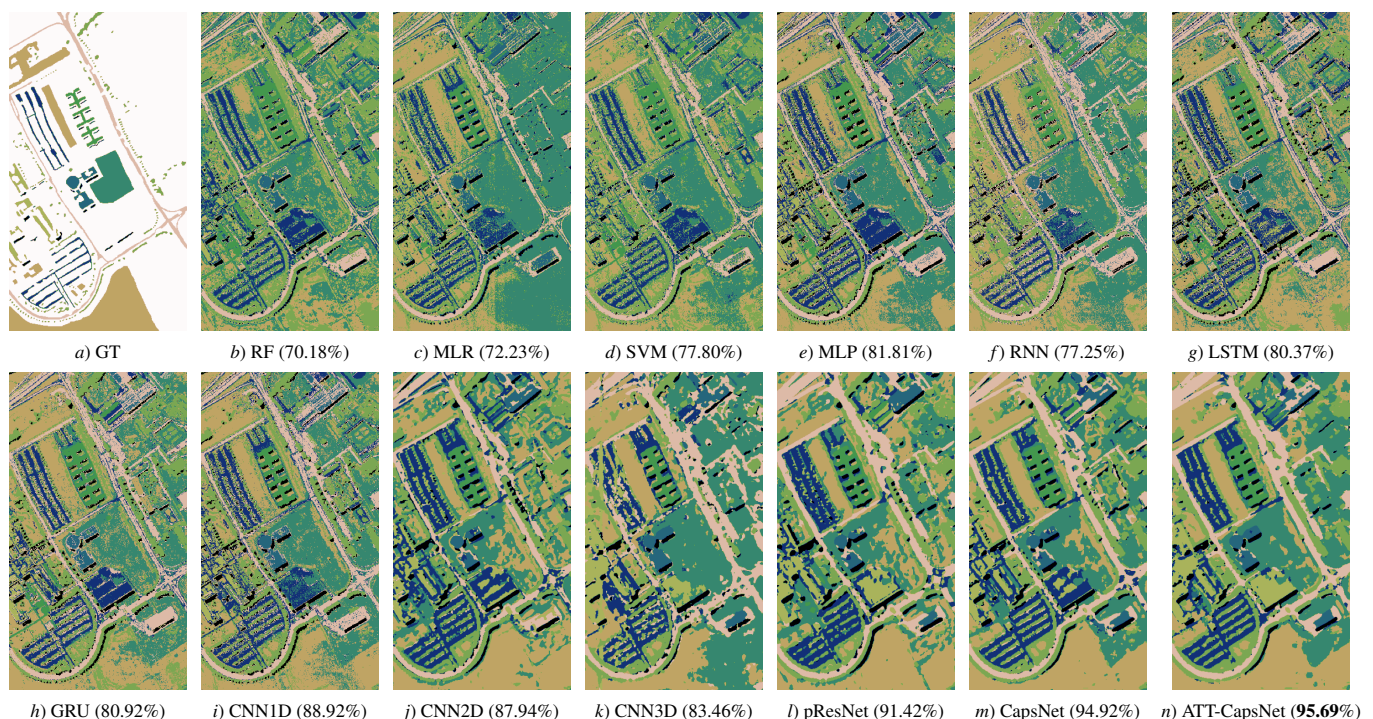


Fig. 10. Visualization of the Ground truth (a) map and the respective classification map for the previous noted methods for the DUP dataset. Accuracy is also included for every method.

multiple attention-guided CapsNet (denoted as *ATT-CapsNet*) achieves the best OA result for IP and UP. Focusing on KSC, the low number of samples negatively affects the model, which suffers strong degradation due to overfitting, whilst in SV (known for the large spectral resemblance between the different lettuce crops), the *ATT-CapsNet* and the pResNet achieve quite similar results, where both attention mechanisms and residual connections play an important role in extracting crucial information for classification. The comparison between the proposed *ATT-CapsNet* and the standard CapsNet is quite interesting. Indeed, the latter reach poor classification results due to its large number of parameters to be adjusted and the weight of the capsules, resulting in a fast degradation because of overfitting problems. Furthermore, the other methods are quite far from the best results obtained by our proposal, the

OA of which is usually around  $\sim 99.xx\%$  of OA for all datasets. Indeed, whilst *ATT-CapsNet*, CapsNet and pResNet are usually on the top of the plots, spectral-based models achieve the worst results, in particular the simpler machine learning models, such as the RF and the MLR. On the contrary, spectral-spatial models such as CNN2D and CNN3D achieve intermediate results.

From Figs. 8a and 8b, in comparison with the other machine and deep learning classifiers, it is quite remarkable the rapid growth in the OA of the *ATT-CapsNet* as more samples are introduced into the model. In fact, a detailed observation of the model has shown a fast slope since the initial iterations.

4) *Accuracy evaluation over disjoint datasets*: Tables V, VI and VII provide the obtained results when conducting HSI classification on the spatially disjoint datasets DIP, DUP and

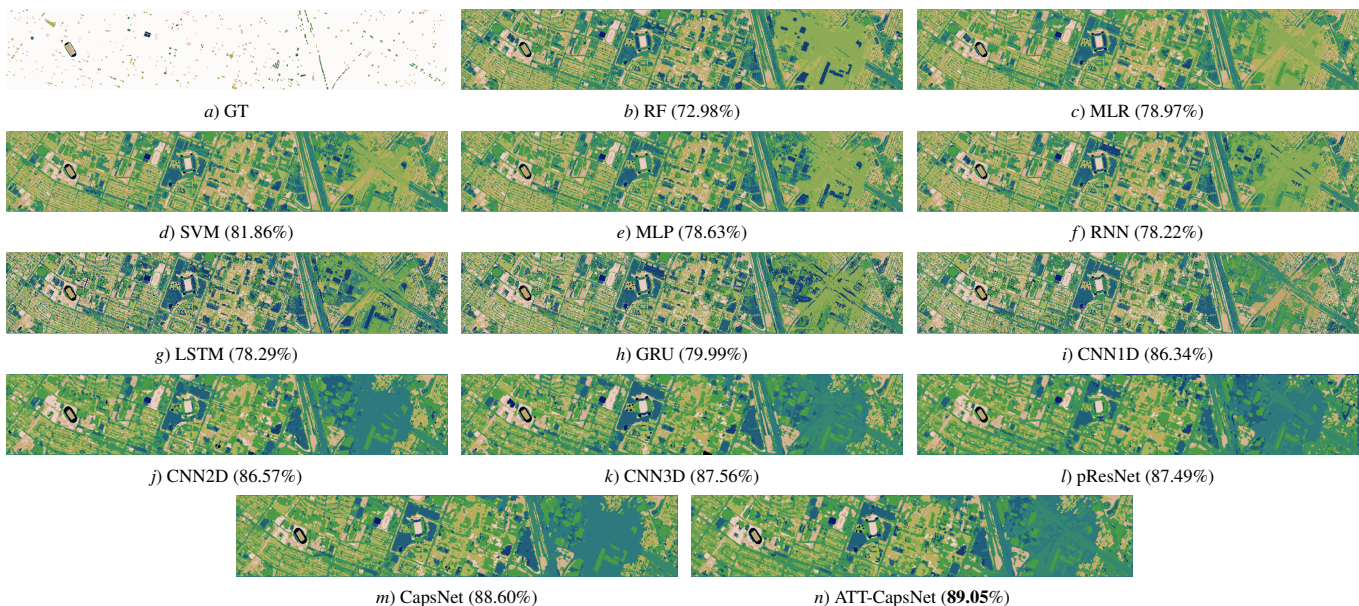


Fig. 11. Visualization of the Ground truth (a) map and the respective classification map for the previous noted methods for the UH dataset. Accuracy is also included for every method.

DUH, respectively. For each classification model, the obtained accuracy-per-class is shown, in addition to OA, AA and K values.

Table V provides the accuracy results over DIP scene. Although there are specific classes in which other models achieve better results (particularly those with few training samples), the proposed *ATT-CapsNet* provides good accuracy-per-class results. The main reason could be the unbalanced classes, for instance the 6th class has a small number of labelled samples. In this context the SVM reaches a good result as it is more robust to high-dimensional data, processing the isolated pixel. As a result, it can produce an accuracy of 100% whilst more complex networks (in particular RNN, GRU, CNN2D, CNN3D, pResNet, CapsNet and *ATT-CapsNet*) are unable to classify these samples. It is interesting to highlight the 3D-GAMO, which thanks to its oversampling technique is able to perform data augmentation of the minority classes, achieving better classification results in them. Nevertheless, and despite these cases, the proposed model provides the best OA (92.27%) and K (91.21%) values.

A similar behaviour can be observed Table VI for DUP, where best global results are obtained by the proposed *ATT-CapsNet* model. Moreover, both *ATT-CapsNet* and CapsNet stand out in comparison to the other machine and deep learning models. In fact, this dataset contains many structures, so that the spatial information has a determining weight in the classification results. In this sense, both capsule-based networks better model this information, extracting relevant spatial relationships between different features. Once more, the *ATT-CapsNet* exhibits the best OA, AA and K results. In particular, its AA is almost 4 points of percentage more than the second best approach (CapsNet) 91.25%.

Also, Table VII provides similar results for DUH scene, where our proposal still gets the best results, although the gap between the *ATT-CapsNet* and the other models is smaller.

For instance, focusing on class 6, the CapsNet achieves an accuracy-per-class of 80.26% percentage points, whilst our methods reaches 80.22%. Furthermore, it can be observed how many proposals manage to obtain a very positive result for some specific classes, which demonstrates the difficulty of achieving an improvement with respect to the literature methods. Nevertheless, the proposed *ATT-CapsNet* provides the best OA, AA and Kappa results, whilst most of its accuracy-per-class values are the highest or close to the highest values. This demonstrates the effectiveness and robustness of the proposed *ATT-CapsNet*, which provides a malleable architecture, which works with different datasets without modifying its architecture and without the resulting classification values varying excessively, obtaining the best results between the current state-of-art methods. As a result, the *ATT-CapsNet* improve the original CapsNet for HSI data classification.

Figs. 9, 10 and 11 depict the classification maps obtained by each model for the DIP, DUP and UH datasets, respectively. The original ground truth values are provided with the best classification map of each method. As we can see, for each dataset the *ATT-CapsNet* reaches the best OA result as we have explained previously in Tables V, VI and VII. Moreover, the proposed model produce clear classification maps.

5) *Impact of spatial information.*: This experiment modifies the input spatial size in order to evaluate the impact of the patch-size on the considered spatial and spectral-spatial models, in particular our proposed *ATT-CapsNet* model is compared with Spectral-Spatial Residual Network (SSRN) [43], DenseNet [46], Dual-Path Network-Based [116], pResNet [44] and CapsNet [63]. Table VIII provides the obtained results in terms of OA for different spatial sizes  $5 \times 5$ ,  $7 \times 7$ ,  $9 \times 9$  and  $11 \times 11$ . In these experiments, where the accuracy is around the values of 98 – 99%, obtaining a profit of a few tenths makes a big difference and is an outstanding improvement in the classification performance of the model.



TABLE VIII  
RESULTS FOR THE THIRD EXPERIMENT REPRESENTING THE OA VALUES IN PERCENTAGE USING MULTIPLE DIFFERENT SPATIAL SIZES FOR DIFFERENT ARCHITECTURES.

		IP dataset				
Spatial Size	SSRN	DenseNet	DPN	pResNet	CapsNet	ATT-CapsNet
5 × 5	92.83 ±0.66	97.85 ±0.28	97.53 ±0.15	98.59±0.24	97.68±0.41	<b>98.90±0.16</b>
7 × 7	97.81 ±0.34	99.24 ±0.14	99.29 ±0.06	99.20±0.15	99.19±0.15	<b>99.59±0.06</b>
9 × 9	98.68 ±0.29	99.58 ±0.09	99.64 ±0.10	99.58±0.11	99.65±0.11	<b>99.71±0.08</b>
11 × 11	98.70 ±0.21	99.74 ±0.08	99.67 ±0.06	99.71±0.10	99.68±0.11	<b>99.75±0.08</b>
		UP dataset				
Spatial Size	SSRN	DenseNet	DPN	pResNet	CapsNet	ATT-CapsNet
5 × 5	98.72 ±0.17	99.13 ±0.08	99.21 ±0.11	99.54±0.07	98.76±0.09	<b>99.56±0.10</b>
7 × 7	99.54 ±0.11	99.71 ±0.10	99.70 ±0.07	99.79±0.04	99.42±0.10	<b>99.85±0.04</b>
9 × 9	99.57 ±0.54	99.73 ±0.15	99.88 ±0.04	99.92±0.04	99.74±0.03	<b>99.92±0.04</b>
11 × 11	99.79 ±0.08	99.93 ±0.03	99.94 ±0.03	99.91±0.04	99.86±0.03	<b>99.96±0.02</b>

In this context, our proposal achieves the best results for all the different most common spatial sizes, obtaining OA values of 98.90%, 99.59%, 99.71% and 99.75% for IP dataset and 99.56%, 99.85%, 99.92% and 99.96% for UP scene.

## V. CONCLUSIONS

In this paper, a multiple attention-driven method based on capsule networks for the classification of HSI datasets is presented. The proposed methodology takes some insights from the attention CNNs state-of-art methods. Specifically, the goal of the implemented *ATT-CapsNet* for HSI data classification is to adapt and implement the capsule networks with different attentions mechanisms in order to extract more refined features at the same time that enhance the routing procedure between capsules, reducing the number of parameters involved. Thus, the *ATT-CapsNet* aims to detect the most relevant features to compress them into capsules invariably to the changes that occur in the input HSIs.

The experimental results show us the effectiveness of the proposed methodology. In addition, the classification maps obtained are quite intuitive. In this regard, the global results and the different types of experiments conducted show a notable improvement over the original CapsNet and other current state-of-art models. The proposed methodology shows great robustness for the classification of most of the different classes for very different datasets. Finally, we consider a great positive factor of our proposal the ability to obtain relevant information at high levels of abstraction, which is a crucial aspect due to the complexity of the HSI data cubes, through the aggregation of characteristics between the child-parent nodes, driven by the self-attention mechanism.

## ACKNOWLEDGMENT

This work has been supported by the computing facilities of Extremadura Research Centre for Advanced Technologies (CETA-CIEMAT), funded by the European Regional Development Fund (ERDF). CETA-CIEMAT belongs to CIEMAT and the Government of Spain.

## REFERENCES

- [1] L. Zhu, J. Suomalainen, J. Liu, J. Hyyppä, H. Kaartinen, and H. Hagren, *Multi-purposeful Application of Geospatial Data*. IntechOpen, 2018, ch. A review: Remote sensing sensors.
- [2] M. B. Stuart, A. J. McGonigle, and J. R. Willmott, "Hyperspectral imaging in environmental monitoring: a review of recent developments and technological advances in compact field deployable systems," *Sensors*, vol. 19, no. 14, p. 3071, 2019.
- [3] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. Nasrabadi, and J. Chanussot, "Hyperspectral remote sensing data analysis and future challenges," *IEEE Geoscience and remote sensing magazine*, vol. 1, no. 2, pp. 6–36, 2013.
- [4] L. Loncan, L. B. De Almeida, J. M. Bioucas-Dias, X. Briottet, J. Chanussot, N. Dobigeon, S. Fabre, W. Liao, G. A. Licciardi, M. Simoes *et al.*, "Hyperspectral pansharpening: A review," *IEEE Geoscience and remote sensing magazine*, vol. 3, no. 3, pp. 27–46, 2015.
- [5] 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, 2019.
- [6] X. Tao, M. E. Paoletti, J. M. Haut, L. Han, P. Ren, J. Plaza, and A. Plaza, "Endmember estimation from hyperspectral images using geometric distances," *IEEE Geoscience and Remote Sensing Letters*, 2021.
- [7] D. N. Kumar and R. Tv, "Remote sensing applications in water resources," *Journal of the Indian Institute of Science*, vol. 93, pp. 163–188, 06 2013.
- [8] C. Weber, R. Aguejdad, X. Briottet, J. Avala, S. Fabre, J. Demuyneck, E. Zenou, Y. Deville, M. Karoui, F. Benhalouche, S. Gadal, W. Ourghemmi, C. Mallet, A. L. Bris, and N. Chehata, "Hyperspectral imagery for environmental urban planning," in *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, 2018, pp. 1628–1631.
- [9] A. C. Schuerger, G. A. Capelle, J. A. Di Benedetto, C. Mao, C. N. Thai, M. D. Evans, J. T. Richards, T. A. Blank, and E. C. Stryjewski, "Comparison of two hyperspectral imaging and two laser-induced fluorescence instruments for the detection of zinc stress and chlorophyll concentration in bahia grass (*paspalum notatum* flugge.)," *Remote sensing of environment*, vol. 84, no. 4, pp. 572–588, 2003.
- [10] C. M. Gevaert, J. Suomalainen, J. Tang, and L. Kooistra, "Generation of spectral-temporal response surfaces by combining multispectral satellite and hyperspectral uav imagery for precision agriculture applications," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 6, pp. 3140–3146, 2015.
- [11] M. J. Khan, H. S. Khan, A. Yousaf, K. Khurshid, and A. Abbas, "Modern trends in hyperspectral image analysis: A review," *Ieee Access*, vol. 6, pp. 14 118–14 129, 2018.
- [12] D. Chutia, D. Bhattacharyya, K. K. Sarma, R. Kalita, and S. Sudhakar, "Hyperspectral remote sensing classifications: a perspective survey," *Transactions in GIS*, vol. 20, no. 4, pp. 463–490, 2016.
- [13] J. M. Haut, M. Paoletti, J. Plaza, and A. Plaza, "Cloud implementation of the k-means algorithm for hyperspectral image analysis," *The Journal of Supercomputing*, vol. 73, no. 1, pp. 514–529, 2017.
- [14] Y. Gu, J. Chanussot, X. Jia, and J. A. Benediktsson, "Multiple kernel learning for hyperspectral image classification: A review," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 11, pp. 6547–6565, 2017.
- [15] P. Ghamisi, J. Plaza, Y. Chen, J. Li, and A. J. Plaza, "Advanced spectral classifiers for hyperspectral images: A review," *IEEE Geoscience and Remote Sensing Magazine*, vol. 5, no. 1, pp. 8–32, 2017.

- [16] 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, 2018.
- [17] M. E. Paoletti, J. M. Haut, X. Tao, J. P. Miguel, and A. Plaza, "A new gpu implementation of support vector machines for fast hyperspectral image classification," *Remote Sensing*, vol. 12, no. 8, p. 1257, 2020.
- [18] Y. Tarabalka, J. A. Benediktsson, and J. Chanussot, "Spectral–spatial classification of hyperspectral imagery based on partitional clustering techniques," *IEEE transactions on geoscience and remote sensing*, vol. 47, no. 8, pp. 2973–2987, 2009.
- [19] O. Rajadell, P. Garcia-Sevilla, and F. Pla, "Spectral–spatial pixel characterization using gabor filters for hyperspectral image classification," *IEEE Geoscience and Remote Sensing Letters*, vol. 10, no. 4, pp. 860–864, 2012.
- [20] M. Fauvel, Y. Tarabalka, J. A. Benediktsson, J. Chanussot, and J. C. Tilton, "Advances in spectral–spatial classification of hyperspectral images," *Proceedings of the IEEE*, vol. 101, no. 3, pp. 652–675, 2012.
- [21] Y. Chen, Z. Lin, X. Zhao, G. Wang, and Y. Gu, "Deep learning-based classification of hyperspectral data," *IEEE Journal of Selected topics in applied earth observations and remote sensing*, vol. 7, no. 6, pp. 2094–2107, 2014.
- [22] C. Chen, W. Li, H. Su, and K. Liu, "Spectral–spatial classification of hyperspectral image based on kernel extreme learning machine," *Remote sensing*, vol. 6, no. 6, pp. 5795–5814, 2014.
- [23] Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio, "Object recognition with gradient-based learning," in *Shape, contour and grouping in computer vision*. Springer, 1999, pp. 319–345.
- [24] Y. LeCun, K. Kavukcuoglu, and C. Faret, "Convolutional networks and applications in vision," in *Proceedings of 2010 IEEE international symposium on circuits and systems*. IEEE, 2010, pp. 253–256.
- [25] L. Zhang, L. Zhang, and B. Du, "Deep learning for remote sensing data: A technical tutorial on the state of the art," *IEEE Geoscience and Remote Sensing Magazine*, vol. 4, no. 2, pp. 22–40, 2016.
- [26] M. Paoletti, J. 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, 2019.
- [27] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.
- [28] H. Gao, S. Lin, Y. Yang, C. Li, and M. Yang, "Convolution neural network based on two-dimensional spectrum for hyperspectral image classification," *Journal of Sensors*, vol. 2018, 2018.
- [29] J. Yue, W. Zhao, S. Mao, and H. Liu, "Spectral–spatial classification of hyperspectral images using deep convolutional neural networks," *Remote Sensing Letters*, vol. 6, no. 6, pp. 468–477, 2015.
- [30] W. Zhao and S. Du, "Spectral–spatial feature extraction for hyperspectral image classification: A dimension reduction and deep learning approach," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 8, pp. 4544–4554, 2016.
- [31] 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, pp. 120–147, 2018.
- [32] 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, 2018.
- [33] L. P. K. Boggavarapu and P. Manoharan, "A new framework for hyperspectral image classification using gabor embedded patch based convolution neural network," *Infrared Physics & Technology*, vol. 110, p. 103455, 2020.
- [34] B. Praveen and V. Menon, "Study of spatial–spectral feature extraction frameworks with 3-d convolutional neural network for robust hyperspectral imagery classification," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 14, pp. 1717–1727, 2020.
- [35] 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, 2018.
- [36] B. Fang, Y. Li, H. Zhang, and J. C.-W. Chan, "Collaborative learning of lightweight convolutional neural network and deep clustering for hyperspectral image semi-supervised classification with limited training samples," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 161, pp. 164–178, 2020.
- [37] S. Dong, Y. Quan, W. Feng, G. Dauphin, L. Gao, and M. Xing, "A pixel cluster cnn and spectral–spatial fusion algorithm for hyperspectral image classification with small-size training samples," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 14, pp. 4101–4114, 2021.
- [38] S. K. Roy, M. E. Paoletti, and S. R. Dubey, "Generative adversarial minority oversampling for spectral–spatial hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, pp. 1–15, 2021.
- [39] D. Hong, L. Gao, J. Yao, B. Zhang, A. J. Plaza, and J. Chanussot, "Graph convolutional networks for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, pp. 5966–5978, 2021.
- [40] Y. Ding, X. Zhao, Z. Zhang, W. Cai, N. Yang, and Y. Zhan, "Semi-supervised locality preserving dense graph neural network with arma filters and context-aware learning for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, pp. 1–12, 2021.
- [41] Y. Ding, X. Zhao, Z. Zhang, W. Cai, and N. Yang, "Multiscale graph sample and aggregate network with context-aware learning for hyperspectral image classification," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 14, pp. 4561–4572, 2021.
- [42] —, "Graph sample and aggregate-attention network for hyperspectral image classification," *IEEE Geoscience and Remote Sensing Letters*, pp. 1–5, 2021.
- [43] Z. Zhong, J. Li, Z. Luo, and M. Chapman, "Spectral–spatial residual network for hyperspectral image classification: A 3-d deep learning framework," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 2, pp. 847–858, 2017.
- [44] 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 Transactions on Geoscience and Remote Sensing*, vol. 57, no. 2, pp. 740–754, 2018.
- [45] Z. Meng, L. Li, X. Tang, Z. Feng, L. Jiao, and M. Liang, "Multipath residual network for spectral–spatial hyperspectral image classification," *Remote Sensing*, vol. 11, no. 16, p. 1896, 2019.
- [46] 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, p. 1454, 2018.
- [47] Y. Bai, Q. Zhang, Z. Lu, and Y. Zhang, "Ssd-cdensnet: A cost-effective end-to-end spectral–spatial dual-channel dense network for hyperspectral image classification," *IEEE Access*, vol. 7, pp. 84876–84889, 2019.
- [48] Z. Niu, G. Zhong, and H. Yu, "A review on the attention mechanism of deep learning," *Neurocomputing*, vol. 452, pp. 48–62, 2021.
- [49] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, "Residual attention network for image classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3156–3164.
- [50] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International conference on machine learning*, 2015, pp. 2048–2057.
- [51] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.
- [52] 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, 2019.
- [53] X. Mei, E. Pan, Y. Ma, X. Dai, J. Huang, F. Fan, Q. Du, H. Zheng, and J. Ma, "Spectral–spatial attention networks for hyperspectral image classification," *Remote Sensing*, vol. 11, no. 8, p. 963, 2019.
- [54] H. Sun, X. Zheng, X. Lu, and S. Wu, "Spectral–spatial attention network for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 5, pp. 3232–3245, 2019.
- [55] L. Mou and X. X. Zhu, "Learning to pay attention on spectral domain: A spectral attention module-based convolutional network for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 1, pp. 110–122, 2019.
- [56] W. Ma, Q. Yang, Y. Wu, W. Zhao, and X. Zhang, "Double-branch multi-attention mechanism network for hyperspectral image classification," *Remote Sensing*, vol. 11, no. 11, p. 1307, 2019.

- [57] Y. Shao, J. Lan, Y. Liang, and J. Hu, "Residual networks with multi-attention mechanism for hyperspectral image classification," *Arabian Journal of Geosciences*, vol. 14, no. 4, pp. 1–19, 2021.
- [58] Z. Dong, Y. Cai, Z. Cai, X. Liu, Z. Yang, and M. Zhuge, "Cooperative spectral-spatial attention dense network for hyperspectral image classification," *IEEE Geoscience and Remote Sensing Letters*, vol. 18, no. 5, pp. 866–870, 2020.
- [59] M. E. Paoletti, J. M. Haut, S. K. Roy, and E. M. Hendrix, "Rotation equivariant convolutional neural networks for hyperspectral image classification," *IEEE Access*, vol. 8, pp. 179 575–179 591, 2020.
- [60] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," 2017.
- [61] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel, "Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness," *arXiv preprint arXiv:1811.12231*, 2018.
- [62] W. Luo, Y. Li, R. Urtasun, and R. Zemel, "Understanding the effective receptive field in deep convolutional neural networks," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 2016, pp. 4905–4913.
- [63] 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, 2018.
- [64] 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.
- [65] J. Yin, S. Li, H. Zhu, and X. Luo, "Hyperspectral image classification using capsnet with well-initialized shallow layers," *IEEE Geoscience and Remote Sensing Letters*, vol. 16, no. 7, pp. 1095–1099, 2019.
- [66] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "Eca-net: efficient channel attention for deep convolutional neural networks, 2020 ieee," in *CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE*, 2020.
- [67] V. Mazzia, F. Salvetti, and M. Chiaberge, "Efficient-capsnet: Capsule network with self-attention routing," 2021.
- [68] B. McIntosh, K. Duarte, Y. Rawat, and M. Shah, "Multi-modal capsule routing for actor and action video segmentation conditioned on natural language queries," *ArXiv*, vol. abs/1812.00303, 2018.
- [69] N. Zhang, S. Deng, Z. Sun, X. Chen, W. Zhang, and H. Chen, "Attention-based capsule networks with dynamic routing for relation extraction," *arXiv preprint arXiv:1812.11321*, 2018.
- [70] Y. Du, X. Zhao, M. He, and W. Guo, "A novel capsule based hybrid neural network for sentiment classification," *IEEE Access*, vol. 7, pp. 39 321–39 328, 2019.
- [71] A. Jaiswal, W. AbdAlmageed, and P. Natarajan, "CapsuleGAN: Generative adversarial capsule network," *ArXiv*, vol. abs/1802.06167, 2018.
- [72] K. Kruthika, Rajeswari, H. Maheshappa, and A. D. N. Initiative, "Cbir system using capsule networks and 3d cnn for alzheimer's disease diagnosis," *Informatics in Medicine Unlocked*, vol. 14, pp. 59–68, 2019.
- [73] R. LaLonde and U. Bagci, "Capsules for object segmentation," *ArXiv*, vol. abs/1804.04241, 2018.
- [74] H. Nguyen, J. Yamagishi, and I. Echizen, "Capsule-forensics: Using capsule networks to detect forged images and videos," *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2307–2311, 2019.
- [75] H. Zhang, L. Meng, X. Wei, X. Tang, X. Tang, X. Wang, B. Jin, and W. Yao, "1d-convolutional capsule network for hyperspectral image classification," *arXiv preprint arXiv:1903.09834*, 2019.
- [76] R. Lei, C. Zhang, S. Du, C. Wang, X. Zhang, H. Zheng, J. Huang, and M. Yu, "A non-local capsule neural network for hyperspectral remote sensing image classification," *Remote Sensing Letters*, vol. 12, no. 1, pp. 40–49, 2021.
- [77] X. Wang, K. Tan, Q. Du, Y. Chen, and P. Du, "Caps-triplegan: Gan-assisted capsnet for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 9, pp. 7232–7245, 2019.
- [78] C. Li, K. Xu, J. Liu, J. Zhu, and B. Zhang, "Triple generative adversarial networks," *arXiv preprint arXiv:1912.09784*, 2019.
- [79] X. Jiang, W. Liu, Y. Zhang, J. Liu, S. Li, and J. Lin, "Spectral-spatial hyperspectral image classification using dual-channel capsule networks," *IEEE Geoscience and Remote Sensing Letters*, vol. 18, no. 6, pp. 1094–1098, 2020.
- [80] J. Wang, S. Guo, R. Huang, L. Li, X. Zhang, and L. Jiao, "Dual-channel capsule generation adversarial network for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, 2021.
- [81] G. E. Hinton, A. Krizhevsky, and S. D. Wang, "Transforming auto-encoders," in *International conference on artificial neural networks*. Springer, 2011, pp. 44–51.
- [82] M. K. Patrick, A. F. Adekoya, A. A. Mighty, and B. Y. Edward, "Capsule networks—a survey," *Journal of King Saud University-computer and information sciences*, 2019.
- [83] E. Xi, S. Bing, and Y. Jin, "Capsule network performance on complex data," *arXiv preprint arXiv:1712.03480*, 2017.
- [84] I. Paik, T. Kwak, and I. Kim, "Capsule networks need an improved routing algorithm," *ArXiv*, vol. abs/1907.13327, 2019.
- [85] J. Rajasegaran, V. Jayasundara, S. Jayasekara, H. Jayasekara, S. Seneviratne, and R. Rodrigo, "Deepcaps: Going deeper with capsule networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 725–10 733.
- [86] M. T. Bahadori, "Spectral capsule networks," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*. OpenReview.net, 2018.
- [87] G. E. Hinton, S. Sabour, and N. Frosst, "Matrix capsules with em routing," in *International conference on learning representations*, 2018.
- [88] D. Wang and Q. Liu, "An optimization view on dynamic routing between capsules," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*. OpenReview.net, 2018. [Online]. Available: <https://openreview.net/forum?id=HJjtFYJdf>
- [89] J. Gu and V. Tresp, "Improving the robustness of capsule networks to image affine transformations," 2020.
- [90] S. Zhang, W. Zhao, X. Wu, and Q. Zhou, "Fast dynamic routing based on weighted kernel density estimation," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 15, p. e5281, 2021.
- [91] N. Zhang, S. Deng, Z. Sun, X. Chen, W. Zhang, and H. Chen, "Attention-based capsule networks with dynamic routing for relation extraction," *arXiv preprint arXiv:1812.11321*, 2018.
- [92] J. Liu, H. Lin, L. Yang, B. Xu, and D. Wen, "Multi-element hierarchical attention capsule network for stock prediction," *IEEE Access*, vol. 8, pp. 143 114–143 123, 2020.
- [93] J. Choi, H. Seo, S. Im, and M. Kang, "Attention routing between capsules," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019, pp. 0–0.
- [94] W. Huang and F. Zhou, "Da-capsnet: dual attention mechanism capsule network," *Scientific Reports*, vol. 10, no. 1, pp. 1–13, 2020.
- [95] A. Hoogi, B. Wilcox, Y. Gupta, and D. L. Rubin, "Self-attention capsule networks for image classification," *arXiv preprint arXiv:1904.12483*, 2019.
- [96] H. Zhao, J. Jia, and V. Koltun, "Exploring self-attention for image recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 076–10 085.
- [97] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.
- [98] Y. Ren, Y. Yu, and H. Guan, "Da-capsnet: A dual-attention capsule unet for road extraction from remote sensing imagery," *Remote Sensing*, vol. 12, no. 18, p. 2866, 2020.
- [99] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.
- [100] S. Ghaffarian, J. Valente, M. Van Der Voort, and B. Tekinerdogan, "Effect of attention mechanism in deep learning-based remote sensing image processing: A systematic literature review," *Remote Sensing*, vol. 13, no. 15, p. 2965, 2021.
- [101] J. Park, S. Woo, J.-Y. Lee, and I. S. Kweon, "Bam: Bottleneck attention module," *arXiv preprint arXiv:1807.06514*, 2018.
- [102] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.
- [103] I. Bello, B. Zoph, A. Vaswani, J. Shlens, and Q. V. Le, "Attention augmented convolutional networks," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 3286–3295.
- [104] T. Alipour-Fard, M. Paoletti, J. M. Haut, H. Arefi, J. Plaza, and A. Plaza, "Multibranch selective kernel networks for hyperspectral image classification," *IEEE Geoscience and Remote Sensing Letters*, vol. 18, no. 6, pp. 1089–1093, 2020.
- [105] H. Gao, M. Wang, Y. Yang, X. Cao, and C. Li, "Hyperspectral image classification with dual attention dense residual network," *International Journal of Remote Sensing*, vol. 42, no. 15, pp. 5604–5625, 2021.
- [106] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.

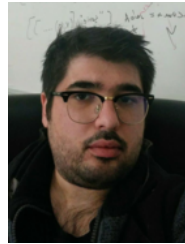
- [107] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015.
- [108] J. Brownlee, "A gentle introduction to the rectified linear unit (relu)," *Machine learning mastery*, vol. 6, 2019.
- [109] P. Baldi and P. J. Sadowski, "Understanding dropout," *Advances in neural information processing systems*, vol. 26, pp. 2814–2822, 2013.
- [110] J. Cheng, L. Dong, and M. Lapata, "Long short-term memory-networks for machine reading," *arXiv preprint arXiv:1601.06733*, 2016.
- [111] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," in *International conference on machine learning*. PMLR, 2019, pp. 7354–7363.
- [112] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.
- [113] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.
- [114] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han, "On the variance of the adaptive learning rate and beyond," in *International Conference on Learning Representations*, 2019.
- [115] B. A. M Graña, MA Veganzons. (2021, Jul.) Gic. hyperspectral remote sensing scenes, grupo de inteligencia computacional de la universidad del país vasco. [Online]. Available: [http://www.ehu.es/ccwintco/index.php/Hyperspectral\\_Remote\\_Sensing\\_Scenes](http://www.ehu.es/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes)
- [116] X. Kang, B. Zhuo, and P. Duan, "Dual-path network-based hyperspectral image classification," *IEEE Geoscience and Remote Sensing Letters*, pp. 1–5, 2018.



**Mercedes E. Paoletti** received the B.Sc and M.Sc. degrees in computer engineering from the University of Extremadura, Cáceres, Spain, in 2014 and 2016, respectively. Also, she obtained her PhD degree in 2020 supported by an University Teacher Training Programme from the Spanish Ministry of Education, as a member of the Hyperspectral Computing Laboratory (HyperComp) at the Department of Technology of Computers and Communications, University of Extremadura. She is currently a researcher at the Department of Computer Architecture and Automation, Complutense University, Madrid. Her research interests include remote sensing and analysis of very high spectral resolution with the current focus on DL and high performance computing. She has served as a reviewer for the IEEE Transactions on Geoscience and Remote Sensing and IEEE Geoscience and Remote Sensing Letters, in which she was recognized as a best reviewer in 2019 and 2020. She was also a recipient of the 2019 Outstanding Paper Award recognition in the IEEE WHISPERS 2019 conference, and a recipient of the Outstanding Ph.D. Award at the University of Extremadura in 2020.



**Sergio Moreno Alvarez** completed his Degree in Computer Engineering (2012-2017) and the Master's Degree in Computer Engineering (2017-2019) at the University of Extremadura. Currently, since October 2019 he is completing his Ph.D in the Department of Computer Systems Engineering and Telematics. As research experience, he has participated regional projects. His main interests are high performance computing, neural networks and DL. He is currently a Researcher in the School of Technology (University of Extremadura). He has published 4 JCR papers in international journals and 3 presentations at international and national conferences.



**Juan M. Haut** is a professor with the Department of Computers and Communications at the University of Extremadura, Cáceres, Spain. Also, he is a member of the Hyperspectral Computing Laboratory (HyperComp) at the Department of Technology of Computers and Communications, University of Extremadura, where he received the B.Sc and M.Sc. degrees in computer engineering in 2011 and 2014, respectively, and the Ph.D. degree in Information Technology in 2019 supported by an University Teacher Training Programme from the Spanish Ministry of Education. Dr. Haut was a recipient of the Outstanding Ph.D. Award at the University of Extremadura in 2019. His research interests include remote sensing data processing and high dimensional data analysis, applying machine (deep) learning and cloud computing approaches. In this sense, he has authored/co-authored more than 40 JCR journal articles (more than 30 in IEEE journals) and more than 30 peer-reviewed conference proceeding papers. Some of his contributions have been recognized as hot-topic publications for their impact on the scientific community. Also, he was a recipient of the Outstanding Paper Award in the 2019 and 2021 IEEE WHISPERS conferences. From his experience as a reviewer, it is worth mentioning his active collaboration in more than 10 scientific journals, such as the IEEE Transactions on Geoscience and Remote Sensing, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, and IEEE Geoscience and Remote Sensing Letters, and he has been awarded with the Best Reviewer recognition of the IEEE Geoscience and Remote Sensing Letters and IEEE Transactions on Geoscience and Remote Sensing in 2018 and 2020 respectively. Furthermore, he has guest-edited three special issues on hyperspectral remote sensing for different journals. He is also an Associate Editor of the IEEE Transactions on Geoscience and Remote Sensing, IEEE Geoscience and Remote Sensing Letters and IEEE Journal on Miniaturization for Air and Space Systems.