A Hybrid Capsule Network for Hyperspectral Image Classification

Massoud Khodadadzadeh, Xuemei Ding, Priyanka Chaurasia, and Damien Coyle, *Senior Member, IEEE*

***Abstract*—Limited training data, high dimensionality, image complexity, and similarity between classes are challenges confronting hyperspectral image (HSI) classification often resulting in suboptimal classification performance. The capsule network (CapsNet) preserves the hierarchy between different parts of the entity in an image by replacing scalar representations with vectors and can address these aforementioned issues. Motivated by CapsNet, this article presents a novel end-to-end deep learning (DL) architecture, the hybrid capsule network (HCapsNet), for HSI classification. HCapsNet employs 2-D and 3-D convolutional neural networks (CNNs) to extract higher level spatial and spectral features. In order to establish a route between capsules in the lower layers to the most-related capsule in the higher layer, dynamic routing (DR) is used to identify several overlapped objects during training sessions. Hyperparameter optimization is performed using nested cross-validation (nested-CV) to ensure thorough generalization evaluation. The proposed HCapsNet significantly outperformed the state-of-the-art methods in terms of overall classification accuracy on three widely used hyperspectral datasets, Indian Pines dataset achieving ($> 3\%$, $p < 1 \times 1 \times 10^{-11}$), the University of Pavia dataset ($> 4\%$, $p < 1 \times 1 \times 10^{-9}$), the Salinas Valley dataset ($> 3\%$, $p < 1 \times 1 \times 10^{-10}$) when using only 1% of the data for training. The performance of all CNN-based approaches degraded significantly with smaller training sample sizes. HCapsNet, therefore, is demonstrated to offer significant advantages in HSI classification problems with low sample sizes.**

***Index Terms*—Capsule neural network (CapsNet), deep learning (DL), dynamic routing (DR), hyperspectral image (HSI).**

I. Introduction

HYPERSPECTRAL data acquisition has considerably increased with the continued advances in imaging technology [1]. Hyperspectral imaging has shown to be a powerful tool for various applications, such as agricultural management [2], [3], environmental protection [4], [5], semiconductor wafer defect detection [6], [7], and mineral exploration [8], [9]. A hyperspectral image (HSI), unlike images used in computer vision, is composed of hundreds of 2-D images corresponding to various spectral bands [10]. This provides a nearly full spectrum of reflected light for each pixel in the image of a scene and thus allows capturing crucial spectral information for materials identification and characterization. Particularly, the remote exploration of the earth's surface is possible with the combination of the spectral and spatial information in these images. In such a framework, HSI classification is undergoing intense study in remotely sensed HSI data analysis [11]. The high dimensionality and lack of sufficient labeled data, large spectrum variability in the spatial domain, and the existence of mixed pixels due to low spatial resolution can, however, decrease HSI classification accuracy. In addition, despite the efforts of experts in processing and evaluating HSI for various applications, due to an ever-growing volume of data, it is critical to developing more intelligent and autonomous approaches.

To date, algorithms applied to HSI classification can be categorized into two general categories: techniques based on manually engineered features and methods based on data-managed features. In the early stage of HSI classification, techniques based on engineered features focused on investigating the influence of spectral features in improving performance. As a result, many pixel classification methods have been presented, including support vector machine (SVM) [12] and multinomial logistic regression [13]. Also, in the techniques based on engineered features, the feature extraction and classifier section are designed independently. For instance, Tang *et al.* [14] proposed a method to reduce the dimension and feature extraction in HSIs using a discrete 3-D scattering wavelet transform. This study examined different benchmarks and showed high accuracies by using a low number of labeled samples. Tatyana *et al.* [15] transformed the HSI into a linear separated space with an active extractor for spectral features, namely regularized linear discriminator. Khodadadzadeh *et al.* [16] combined the classic multinomial logistic regression (MLR) formulation with a class-dependent subspace projection method to cope with highly mixed hyperspectral data using limited training samples. In the literature, several techniques have been applied to perform supervised classification of hyperspectral data [17]. In [12], SVMs and nonparametric classifiers are compared for a multiclass classification task. It was concluded that radial basis function RBF-SVM is more efficient compared with linear-SVM, k-nearest neighbors (KNN), and other RBF kernel methods. Two classifiers based on the random forest (RF) methods were investigated in [18] to enhance generalisation in HSI classification. This study showed that applying RF ensembles instead of a single tree, enhance

Manuscript received July 26, 2021; revised October 1, 2021 and October 25, 2021; accepted October 31, 2021. Date of publication November 9, 2021; date of current version December 1, 2021. This work was supported in part by Tier 2 High Performance Computing resources provided by the Northern Ireland High Performance Computing (NI-HPC) facility funded by the U.K. Engineering and Physical Sciences Research Council (EPSRC) under Grant EP/T02217, in part by the UKRI Turing AI Fellowship 2021–2025 funded by the EPSRC under Grant EP/V025724/1, and in part by Vice-Chancellor's Research Scholarship (VCRS). *(Corresponding author: Massoud Khodadadzadeh.)*

The authors are with the School of Computing, Engineering and Intelligent Systems, Ulster University—Magee Campus, BT487ET Derry, U.K. and Northern Ireland (e-mail: khodadadzadeh-m@ulster.ac.uk; x.ding@ulster.ac.uk; p.chaurasia@ulster.ac.uk; dh.coyle@ulster.ac.uk).

Digital Object Identifier 10.1109/JSTARS.2021.3126427

classification accuracy. However, traditional machine learning methods commonly face challenges due to the complexity in HSI dataset characteristics, including the nonlinear relationship between elements and their respective spectral information [19]. Moreover, while these machine learning methods have been applied and evaluated, the interactions between the classifiers and feature extraction approaches are rarely considered.

On the other hand, methods based on data-managed features involve extracting features during the learning stage. These methods use DL structures. DL-based structures have gradually dominated remote sensing image scene classification as DL theory, and parallel computing resources have improved [20]. Hinton *et al.* [21] designed a method for initialising the weights of multilayer neural networks, laying the groundwork for the later development of DL such as stacked autoencoder [22], autoencoder [23], and deep belief nets [24]. These aforementioned methods commonly use convolutional neural networks (CNNs). In CNNs with a series of matrix multiplications (kernels), the mapping between the input data and the output label is performed, e.g., in [25], a deep contextual CNN approach is proposed. This study suggested a wide and deep CNN architecture that employs a combination of state-of-the-art methods such as GoogLeNet [26] and ResNet [27] and achieved higher classification accuracy than a shallower network. To extract spectral and spatial features from HSI effectively, Chen *et al.* [28] proposed an end-to-end 3-D-CNN that provides significant accuracy when the training samples are limited. Li *et al.* [29] presented a new CNN-based DL structure for classification, which outperformed commonly used algorithms. In this method, instead of end-to-end convolution and deconvolution layers, an optimized extreme learning machine (ELM) after-layers is employed. An overview of the shallow and deep techniques with an advanced feature extraction approach is detailed in [30]. As stated in [30], the lack of sufficient training data in the remote sensing community generally requires the use of feature extraction in both machine learning and DL techniques to overcome this problem. Autoencoders (AEs) and recurrent neural networks (RNNs) must vectorize the inputs during spatial feature extraction, despite the fact that CNNs are genuinely good at spectral-spatial input processing. Therefore, generally, the integration of these networks can deliver the full advantage of their various benefits. This has been done in stacked convolutional AE (SCAE) [31] and convolutional RNN (CRNN) [32], where spectral-spatial joint features extraction was proposed. Also, studies showed that employing only a 2-D-CNN may result in missing data on channel relationships (spatial information), while using only 3-D-CNN may result in a highly complex model [33]. Therefore, HSI applications may be thoroughly investigated with both spectral features and spatial patterns (2-D-CNN and 3-D-CNN), and classification performance can be substantially enhanced. In [34], an architecture called hybrid spectral convolutional neural network (HybridSN) was designed using a combination of 2-D and 3-D CNNs, which has achieved higher accuracies compared to the 3-D-CNN on benchmark datasets.

It is demonstrated in [35] that nonconvex modeling and optimization is a powerful tool that can be applied to various areas. This allows for the development of new techniques and the implementation of interpretable artificial intelligence (AI) for various hyperspectral remote sensing applications. Despite the effectiveness of DL in single-modality-dominated classification tasks, Hong *et al.* [36] introduced a multimodal DL structure intending to provide a baseline solution for pixel-level remote sensing image classification problems using multimodal input. This study uses fully connected networks (FC-Nets) and CNNs, which can apply to pixel-based and spatial-spectral classification, respectively. Graph convolutional networks (GCNs) have a high computational cost, especially noticeable in large-scale remote sensing problems. To address this, a new supervised version of GCNs called miniGCNs is proposed in [37] that can properly characterize the underlying data structure of HS images in high-dimensional space. Although it was hypothesized that deeper CNNs might improve performance and produce better hyperspectral feature representation, the vanishing gradient problem can occur in the model, and consequently, failure in parameter convergence and overfitting may result in the scenario of limited training data [38]. In order to mitigate the disadvantage of using a deeper network which may decrease the accuracy due to vanishing gradient in the model, spectral-spatial residual network (SSRN) [39] employs different residual blocks between layers. These blocks collected abundant spectral and spatial features in the model.

What is evident from the literature is that there exist a number of constraints in CNNs, including the invariance generated by pooling and their inability to determine the spatial relationship between features due to several fully connected layers appended to the final layers. In addition, most of those models require large amounts of labeled data and many iterations to train. The high cost of labeling severely limits their scalability to new categories. Furthermore, this limits their applicability to a small number of scene types (e.g., military zones), which are difficult to capture. Humans, on the other hand, are capable of distinguishing scenes with little or no supervision learning [40], [41]. In other words, children can identify TV scene types based on a single image or image description. To date, the most advanced scene classification methods still fall far short of what humans are capable of doing with only a handful of labeled samples [20].

In the capsule network (CapsNet), these limitations are addressed [42]. In recent years, CapsNet has been proposed as an alternative and successful approach to DL. Instead of traditional scalar points, in CapsNet, vectors facilitate the characterisation of the relationships between the information available in the features and identify more attributes to enable class discrimination [43]. CapsNet has been employed in several research areas such as electroencephalogram (EEG) classification [43], MRI image classification [44], object detection [45], image segmentation [46], and remote sensing [47], although it is still in the early stages of development. For HSI classification, CapsNet has been employed in [48] and compared against CNN methods. The result demonstrated that the capsule-based architecture, named CAP, can provide high overall accuracies on benchmark datasets with higher complexity. Zhang *et al.* [49] proposed an architecture by combining CNN and CapsNet to exploit the benefits of both models. In their model, a DL-based feature extractor is trained on the ImageNet dataset [50] after which the feature-map is fed into the new CapsNet model, which is designed for HSI classification. The results demonstrated that the pretrained model on Inception-v3 [51] gained an overall classification accuracy higher than VGG-16 [52]. Several studies have demonstrated that CapsNets are able to address the problem of hyperspectral mixed pixels classification. When an object in HSI is smaller than the spatial resolution, neighborhood mixing occurs. While a large number of training samples are required in DL networks to obtain reliable trained parameters and prevent overfitting, CapsNets can be trained from a limited training data, which is a common constraint in HSI [53].

There are some challenges for the existing capsule-based networks for HSI classification. First, the feature extraction section is poorly designed, and it is not considered that this part should include both spectral and spatial data in a single classification framework (for example, in [45] and [48]). As a result, the extraction of features using spectral and spatial patterns (2-D-CNN and 3-D-CNN) may be investigated entirely. Since CapsNet is not designed for HSI classification, we cannot adopt the original structure. For example, in [54], CAPSNET operated the original architecture of the CapsNet on HSI datasets with a shallow structure of two 2-D-CNN for organizing the primary capsules. Second, some capsule-based networks for HSI classification tried to show how their model can deal with a limited number of parameters; however, deep learning studies with small test and train sample sizes require a careful validation process. Without nested cross-validation (nested CV), the same data are used to tune model parameters and evaluate model performance in a model selection. As a result, information may leak into the model, causing the data to overfit [55]. For example, in [49] and [54] simple cross-validation is performed for CapsNet-base models. Here, we promote and demonstrate the use of nested-CV for improved hyperparameter optimisation and generalization.

The motivation of this study is to propose a novel end-to-end DL architecture involving a hybrid of CNN and CapsNet (Hybrid CapsNet - HCapsNet) with robust feature learning while using a limited number of training samples. In addition, HCapsNet addresses the issue of high dimensionality and class similarity for HSI classification. Instead of the traditional max-pooling process, HCapsNet employs dynamic routing (DR) as a novel routing method in the CNN architecture and combines 2-D and 3-D CNNs to extract higher level spatial and spectral features. These features are, unlike the traditional structure of the neurons, vectors. This information vector transformation preserves the entity's precise posture feature information. Following that, features are reshaped and employed as an input vector to the next layer for DR. Finally, the decoder part, which acts as a regularizer, is added. Since we dealt with limited training samples, Nested-CV is applied for hyperparameter optimization, and the best parameters for the proposed architecture are reported. The proposed architecture is evaluated using three widely used hyperspectral benchmark datasets collected by the airborne visible infrared imaging spectrometer (AVIRIS) over the Indian Pines (IP), Indiana (16 classes), and Salinas Valley (SV), California (16 classes), and by the reflective optics spectrographic imaging system (ROSIS) over the city of Pavia (UP), Italy (9 classes). The method is benchmarked against SVM [12], 2-D-CNN [56], 3-D-CNN [57], HybridSN [34], SSRN [39], SCAEs [31], CRNNs [32], and CAPSNET [54]. The major contributions of the article can be summarized as follows.

1) An efficient hybrid CapsNet is proposed, which can be applied for the classification of HSIs with a low number of training samples.
2) A DR technique inspired by CapsNet is incorporated in our model, which significantly improves the extraction of spatial-spectral features.
3) A nested-CV technique is applied to find the best model parameters. To the best of our knowledge, this is the first study exploring the nested-CV for this purpose and is recommended approach for thorough evaluation of generalisation performance.
4) A thorough experimental comparison of the presented approach with other recently proposed advanced DL techniques using three available benchmark datasets.

The rest of this article is organized as follows. Section II explains methods (i.e., CNNs, CapsNet approach, and a general overview of DR in terms of structure, routing prediction, and the procedure). Following that, the proposed HCapsNet and analysis are described. The experimental results on different HSI benchmarks, computational time, hyperparameter selection, and ablation study are provided in Section III. The results are discussed in Section IV. Finally, Section V concludes this article.

II. Methods

In this section, both 2-D and 3-D CNN approaches in terms of the network structure, operations, kernels dimension, and blocks arrangements which are later used in the proposed HCapsNet architecture are explained. In addition, a comprehensive representation of the CapsNet structure with the idea of neurons vectorization is elaborated. More importantly, the DR algorithm, which is applied to make the feature prediction between capsule layers as the principal part of the CapsNet, is explained. Next, the description of the proposed method and the datasets is described. Additionally, statistical analysis is defined.

A. CNNs Approach

The objective of a CNN is to learn how the input data and the output data are mapped [58]. As the HSI datasets are volumetric, when designing feature maps, employing 2-D-CNN and 3-D-CNN make it possible to achieve the highest accuracy in the model. For a 2-D-CNN, a series of matrix multiplications named kernel filters with a sample size (P_i, Q_i) in the ith layer are applied followed by a summation operation to the input images. The objective is to detect parts of that image with the most relevance [59]. To elaborate, V_{ij}^{ab}, the output centered at (a, b) for the jth feature map and the ith layer can be expressed as

$$V_{ij}^{ab} = \tanh\left(b_{ij} + \sum_{m} \sum_{x=0}^{P_i-1} \sum_{y=0}^{Q_i-1} W_{ijm}^{xy} V_{(i-1)m}^{(a+x)(b+y)}\right) \quad (1)$$

where the hyperbolic tangent $(tanh)$ is the nonlinearity operation employed on the kernel output and biases (b_{ij}). W_{ijm}^{xy} and m are the kernel output centered at (x, y) for the kth feature map and the indexing parameter over a feature maps set connected to the current feature map in the $(i-1)$th layer, respectively. In general, the output of the convolutional layer is a feature map, which represents several features learned from the input image. The convolutional block is concluded by a pooling or subsampling operation. In the pooling operation, sections are created for all pixel values, but only the maximum pixel value is obtained from each section. Finally, the high-level reasoning is carried out using layers that are fully connected, following several layers performing convolution and max pooling. In a fully connected layer, neurons are connected to every activation in the previous layer. Basic features such as edges and bright spots are learned in the convolutional network's lowest layer. Once these features are learned, more complex shapes and patterns are learned across multiple layers to enable the convolutional networks to classify images.

Similarly, considering that R_i is the third dimension of the kernels in 3-D-CNN, the output centered at (a, b, c) for the jth feature map and the ith layer is formulated in (2), where W_{ijm}^{xyz} is the kernel output centered at $(x, y\, z)$ for the kth feature map and m indicates feature maps indexes

$$V_{ij}^{abc} = \tanh \times \left(b_{ij} + \sum_{m} \sum_{x=0}^{P_i-1} \sum_{y=0}^{Q_i-1} \sum_{z=0}^{R_i-1} W_{ijm}^{xyz} \, V_{(i-1)m}^{(a+x)(b+y)(c+z)} \right) \tag{2}$$

B. CapsNet Approach

In terms of the structure of the neurons, the architecture of the network, propagation, and interlayer distribution methods, there is a significant difference between CapsNet and traditional CNN models [42]. The capsule neuron is a critical formation in CapsNet. The parameters used as input and output in CapsNet are, in contrast with the traditional structure of the neuron, vectors. Hence, the inside parameters correspond to vectors, with differences in the employed activation functions. To elaborate, the neuron input and output vectors in the capsule indicate the parameters employed for the instantiation of a particular type of entity i.e., the vector length represents the probability for entity existence, and the vector direction indicates the presence of the entity attribute.

In a general description, capsule networks are networks designed to obtain inverse graphics. Capsules receive an image and identity of its containing objects as well as their instantiation parameters [42]. Therefore, capsules are defined as functions capable of predicting instantiation parameters for any designated object at a specific location. The estimated probability of an object is represented via the length of an activation vector. Also, the activation vector's orientation reveals the instantiation parameters of the object. Since the CapsNet is robust to affine transformations and can interpret them, the instantiation parameters can be rotation, skewed, stretched, thickness, etc. Since the length of the vectors represent a probability that should be less than or equal to 1, a squashing function can be utilized. In each layer, the capsule's objective is to anticipate the output of the subsequent layers according to the previous layer. In order to perform such a prediction, the dot product is employed. The hierarchy is obtained easily through monitoring the activation pathways and comprehend the parts that belong together with high precision.

The structure proposed for CapsNet is straightforward. The model for the original CapsNet, as demonstrated in Fig. 1, includes various layers, namely convolution layers, primary capsule, DigitCaps (second capsule), and fully connected layers. A handwritten digit image is fed into the model and each layer comprises a convolution (Conv1 and Conv2 with the same kernels and different strides). The rectified linear unit (ReLU) is used to activate the classic convolution layers in the convolution layers. As a consequence, Conv1 and Conv2 produce different feature maps. The primary capsule layer is formed by reshaping the feature maps and is responsible for constructing the corresponding vector structure and acts as the capsule input layer. Moreover, this layer is responsible for reshaping, which is then applied as an input vector to the following layer.

Similarly, the DigitCaps layer represents the output layer for the capsule. The loss function for the classification objective (encoder part) is determined after the DigitCaps, while the fully connected layers are to reconstruct the images (decoder part) to act as network regularizations that prevent overfitting. The DR algorithm is employed between the primary capsules and the DigitCaps to update the calculations and the parameters necessary between the full connections. As for updating the parameters, in CapsNets, DR is employed along with the traditional back-propagating method [42].

C. Dynamic Routing Algorithm

As in (3), the primary capsule u_i $(i = 1, \ldots, 1152)$ in the DR algorithm is multiplied by a weighted matrices W_{ij} to predict the next level capsule output $\hat{u}_i^j$ in the next ten (since the network is going to classify 10 digit numbers 0 to 9) different DigitCaps $(j = 1, \ldots, 10)$. The 16-D capsules output $\hat{u}_i^j$ (4) is determined by multiplying the weighted matrix multiplication W_{ij} and 8-D primary capsules u_i. Therefore, the dimension of W_{ij} is $[16 \times 8]$. In other words, the transformation matrix in the same capsule level speculates the instantiation parameters of the capsules at higher levels

$$\hat{u}_i^j = w_{i,j} \, . \, u_i \tag{3}$$

$$\begin{bmatrix} \hat{u}_i^j(1) \\ \vdots \\ \vdots \\ \hat{u}_i^j(16) \end{bmatrix}_{16\times 1} = \begin{bmatrix} w_{i,j}(1) & \cdots & w_{i,j}(8) \\ \vdots & \ddots & \vdots \\ \vdots & & \vdots \\ w_{i,j}(120) & \cdots & w_{i,j}(128) \end{bmatrix}_{16\times 8} \cdot \begin{bmatrix} u_i(1) \\ \vdots \\ u_i(8) \end{bmatrix}_{8\times 1} . \tag{4}$$

To elaborate, calculations for the first capsule $(i = 1)$ and the last $(i = 1152)$ are shown in the following:

$$\hat{u}_1^1 = w_{1,1} . \, u_1 \quad \ldots \quad \hat{u}_{1,152}^{10} = w_{1152,10} \, . \, u_{1152} . \tag{5}$$

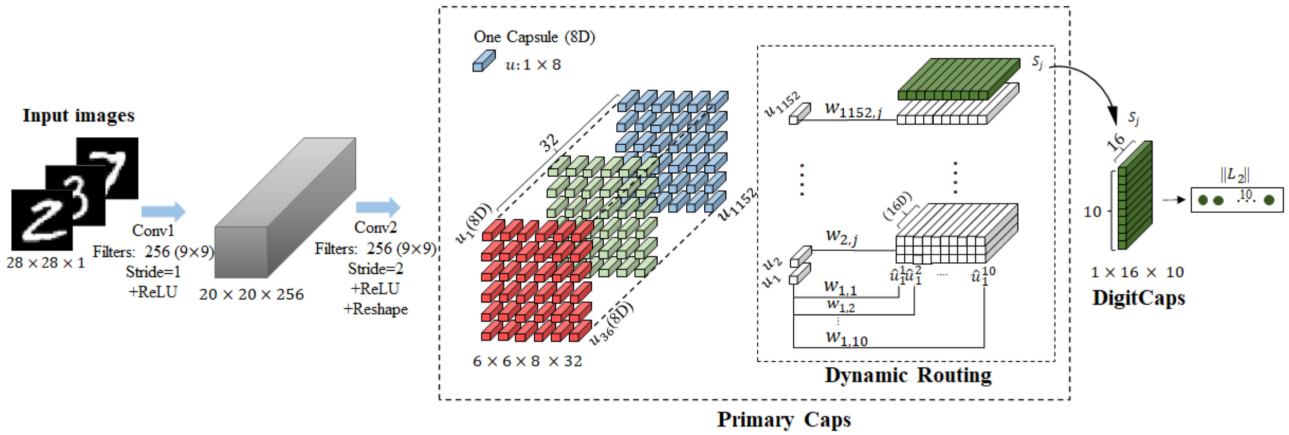


Fig. 1. Structure of the CapsNet (encoder). The model receives as an input image a handwritten digit (MNIST database of handwritten digit [60]) with size $28{\times}28{\times}1$ and learns to encode it into a $16D$ vector of instantiation parameters (DigitCaps) for 10 different classes. The Convolution layers are classic convolution layers with a rectified linear unit (ReLU) activation function to implement the extraction of local features. There are two convolutional layers (Conv1 has 9x9 convolutional kernel with a stride of 1 and ReLU and Conv2 has 9×9 convolutional kernel with a stride of 2). Conv1 results in 256 20×20 features maps and the second results in 256 6×6 (32×8 × 6×6) feature maps which contain scalars. This output is reshaped to get 32 6×6 maps, including 8-D vectors. In total, we have 1152 capsules resulting in a list of 11 520 predictions: 1152 prediction × 10 classes = 11520 weighted matrices W_{ij}.

These calculations are performed with element-wise matrix multiplications, where all matrices hold the same dimensions $[1152 \times 10]$.

The process for the algorithm in CapsNets, with the presence of a routing agreement, can be seen in Algorithm 1. In the training session, one batch of the input image at each iteration is fed to the input layer. Next, following two convolutional layers and reshaping, $\hat{u}_i^j$ is performed. Then, at the first iteration $(r = 1)$, the log probability b_{ij} (i.e., the primary capsule i should be transmitted to DigitCaps j) is set to 0 (raw routing weight) for each predicted output $\hat{u}_i^j$. After that, the softmax function as follows is implemented on these raw weights for each primary capsule:

$$c_{ij} = \frac{\exp(b_{ik})}{\sum_k \exp(b_{ik})}. \tag{6}$$

Next, a weighted sum of the predictions is measured (7), for each capsule in the subsequent layer, along with applying the squashing function (8). The squashing function, which compresses and non-linearizes the vectors, performs the capsule neuron's activation function. In other words, this activation function includes two-parts: the unit length (right part) that is responsible for preserving the length of the vectors between 0 and 1, and the additional scaling (left part) that is responsible for preserving the direction of the vectors

$$s_j = \sum_i c_{ij}\ \hat{u}_{j|i} \tag{7}$$

$$v_j = \frac{||s_j||^2}{1 + ||s_j||^2} \frac{s_j}{||s_j||}. \tag{8}$$

Next, by applying the dot product (9), the prediction (strong agreement) between the lower and upper capsule is performed. After mostly two or three iterations, the routing weight demonstrated in (10) is updated

$$\hat{u}_{j|i} .\ v_j \tag{9}$$

Algorithm 1: Training and Backpropagation.

```
for numbers of epochs do
    for iterations (batches) do
        for numbers of routing(r) do
            $b_{i,j}\ (initialized)^{(r=1)} = 0$
            $b_{i,j} \ \rightarrow c_{i,j}$ ▷ Coupling coefficient (Softmax)
            $S_j\ ^{(r)} = \sum_i c_{i,j}\ .\hat{u}_i j^{(r)}$   ▷ Weighted sum
            $V_j\ ^{(r)} = Squash(S_j{}^{(r)})$   ▷ Product vector
            $= Squash(\sum_i c_{i,j}\ .\hat{u}_i j^{(r)})$
            Dot Product: $\hat{u}_i^j\ .\ V_j\ ^{(r)}$
            $b_{i,j}{}^{Updated} \leftarrow b_{i,j} + \hat{u}_i^j\ .\ V_j\ ^{(r)}$
                $\leftarrow b_{i,j} + \hat{u}_i j\ ^T\ .\ V_j\ ^{(r)}$
              $\leftarrow b_{i,j} + \hat{u}_i j\ ^T\ .\ Squash(\sum_i c_{i,j}\ .\hat{u}_i j^{(r)})$
        return $V_j$ (end for routings)
        The loss function for the correct category:
        $L_J = k.\ max(0,\ 0.9 - ||V_j||)$
        Backpropagation starts:
        Weighted matrices: $w_{i,j}{}^{updated} \leftarrow\ w_{i,j}$
        Convolution layers: $filters^{updated} \leftarrow filters$
    end for (iterations)
end for (epochs)
```

$$b_{ij} = b_{ij}\ + \hat{u}_{j|i}\ v_j \tag{10}$$

Hence, in the top layer, an image classifier can be designed with one capsule per class. Then, it is necessary to append a measuring layer to calculate the length of the top layer activation vectors, which obtains the estimated class probabilities. In standard classification neural networks, training is carried out via minimising the cross-entropy loss. However, Hinton [42] applied margin loss L_k (11) to enable detecting multiple classes in the image. According to this equation, if an object, within class k, can be detected in the image, the output for the corresponding capsule length in the top-level should be more than or equal to

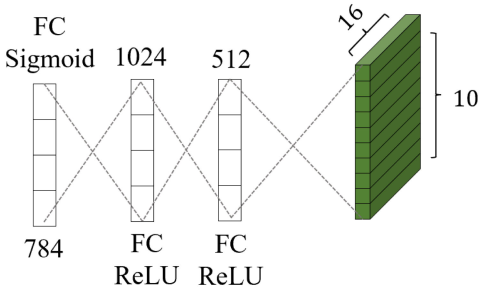


Fig. 2. Reconstruction in CapsNet (Decoder). The decoder contains three fully connected layers with the number of neurons 784, 1024, 512.

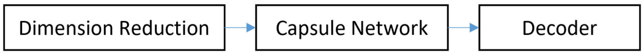


Fig. 3. Proposed HCapsNet block diagram.

0.9. On the other hand, if such an object cannot be detected, then the output of the capsule should be a short vector with a length of less than 0.1

$$L_k = T_k \max(0, m^+ - ||v_k||)^2 + \lambda(1 - T_k) \max(0, ||v_k|| - m^-)^2. \quad (11)$$

To allow multiple classes, the margin loss $T_k = 1$ is minimized if and only if class k is present. In this equation, $m^- = 0.1$, $m^+ = 0.9$ and $\lambda = 0.5$. Once the routing weights ${b_{i,j}}^{Updated}$ are updated, the backpropagation will initiate. Using the Adam optimizer [61] in TensorFlow, by agreement in this stage, the gradient of loss in the network is generated, and weighted matrices and filters in convolution layers are updated without route checking. After this step is concluded, we can proceed to the next batch of image sets.

In the original CapsNet after three fully connected layers are derived, they are combined as a decoder network to the top of the CapsNet (see Fig. 2).

To learn the input image reconstruction, the squared difference between the reconstructed image, and the input image should be minimized. The overall loss (12) is the summation of the margin loss and the reconstruction loss. To ensure that the dominating loss in the training stage is the margin loss, the reconstruction loss is scaled down with a significantly small coefficient ($\alpha = 0.0005$)

$$Loss = margin\ loss + (\alpha.reconstruction\ loss). \quad (12)$$

D. Proposed Hybrid Capsule Neural Network

Here, we explained the formation of our proposed architecture. Although architectures in computer vision with deeper layers can yield better feature extraction, this is not the case in HSI and, often model performance will be decreased with deeper layers [39]. In the original CapsNet, since the model is designed for conventional computer vision images, it cannot be applied directly to HSI. Hence, to obtain a suitable model for HSI classification, the leading workflow of the proposed HCapsNet is divided into the following three sections (see Fig. 3).These three sections are explained below.

1) Dimension Reduction: First, principal components analysis (PCA) is applied on labeled dataset to decrease spectrum redundancy. It compresses the HSI that reduces the number of spectral bands. Then, without feature engineering, the corresponding image is divided into patches labeled with respect to the central pixel. Finally, 2-D–3-D CNNs are implemented to extract the spectral-spatial feature map (see Fig. 4).

2) Capsule Network: In this step, the output is first reshaped into nD vectors $u_i (i = 1, \ldots, I)$ as a vector length n for a single capsule in the CapsNet structure. Then, it is multiplied by weighted matrices W_{ij} to predict the capsules for the next level (called ClassCapsule $\hat{u}_i^j (j = 1, \ldots, J)$), where I and J are the number of primary capsules and classes, respectively. Here, m is the vector length for one ClassCapsule. Then, as explained in Section II, the weighted sum S_j will be calculated, followed by the squash function. After that, the DR process will be performed to send the most related capsule in PrimaryCaps to the ClassCapsule. Finally, a classifier can be designed in the top layer with one capsule per class. It means that the characteristics and feature spatial relationships are encoded correctly (see Fig. 5).

3) Decoder: In the decoder, an additional reconstruction loss is applied to drive the next level capsule called ClassCapsule to encode the instantiation parameters from input data. Although the decoder functions similarly to a regulariser, adding the margin loss to prevent overfitting during training, it is not dominant in the margin loss because of the scaling factor applied (0.0005).

Tables I and II show the final selected architecture for the datasets used. In this case, to show the structure of different layers with different input and output shapes, 25 principal components (PCs) are selected for the IP dataset and 15 for the SA and UP datasets. The 3-D-CNN and 2-D-CNN are employed to form the output. After that, the output is reshaped into capsules (PrimaryCaps) which is then sent to the most relevant ClassCapsule in the next layer by the DR algorithm for the classification. The decoder is also added to the final stage of the model. Dataset and analysis description are described in subsections below. Also, the effect of different number of PCs is analyzed in the following section.

E. Data Description

To evaluate and compare the performance of the proposed method, three widely-used HSI datasets, (i.e., Indian Pines (IP), Salinas Valley (SV), and University of Pavia (UP) [62]) were employed in this research. The dataset statistics and information regarding different environmental frameworks are described in Table III.

All the datasets are obtained using aerial sensors. HSI is very well known to be prone to atmospheric conditions, which can cause difficulty in processing. Therefore, applying band selection and normalization is necessary. Particularly, spectrum dynamics transform due to sensors conditions. Therefore, to enhance classifier robustness, water absorption band and low signal to noise ratio bands are excluded. Moreover, PCA and mutual information can remove uninformative bands and reduce spectrum redundancy [63], [64]. There are 16 classes in IP and SV with the total number of the samples 10 249 and

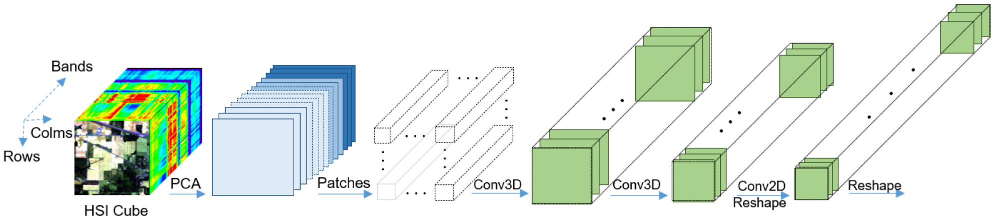


Fig. 4. Feature extraction in the proposed architecture. As demonstrated here, a hyperspectral cube dimension is reduced following PCA, patch windowing, and CNNs. All the parameters used for dimension reduction is completely explained in the hyperparameters settings.

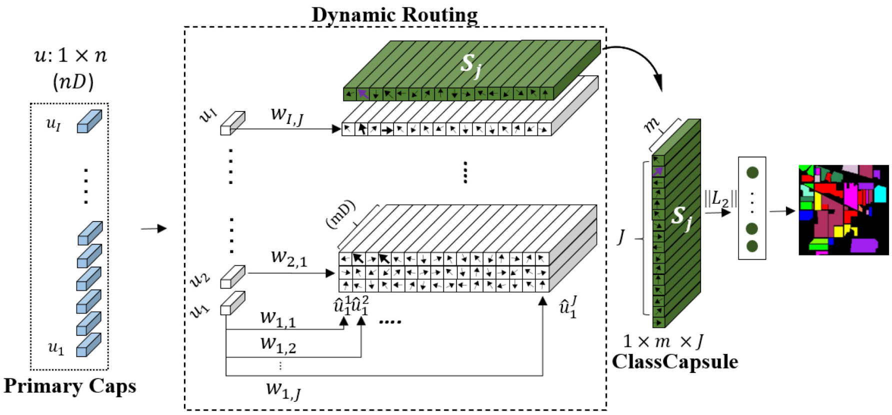


Fig. 5. Capsule network in the proposed HCapsNet architecture. The different arrows demonstrated a vector with different length and orientation, which means different parts of the feature are identified in that specific part of the data. The number of the Primary Caps (I) depends on the dimension of the feature map in the last convolution layer-the number of the ClassCapsule (J) is equal to the number of the classes in the specific dataset.

TABLE I
SUMMARY OF DIFFERENT LAYERS EMPLOYED IN THE PROPOSED HCAPSNET FOR 25 × 25 WINDOW SIZE WITH PCS = 25 FOR IP DATA PATCHES

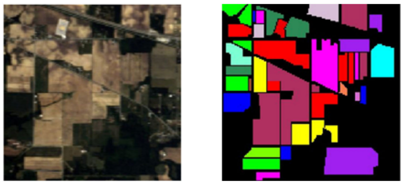
Datasets	Layers	Shape	Output	Stride	BN	Padding	Activatin Function	Parameters
	Input Layer	-	(25, 25, 25, 1)	-	-	-	-	0
	Conv3D-1	(9, 9, 7)	(17, 17, 18, 8)	1	Yes	No	LeakyReLU	4544
	Conv3D-2	(9, 9, 5)	(9, 9, 16, 16)	1	No	No	LeakyReLU	51856
IP	Conv2D	(3,3)	(7, 7, 64)	1	Yes	Yes	LeakyReLU	184384
	PrimaryCaps & reshape & squash	-	(392, 8)	-	-	-	LeakyReLU & Squash	0
	ClassCapsule	-	(16, 16)	-	-	-	-	802816
	Mask	-	(256)	-	-	-	-	0
	Decoder	-	(25, 25, 30, 1)	-	-	-	-	19875646
								Total trainable parameters: 20,919,246

TABLE II
SUMMARY OF DIFFERENT LAYERS EMPLOYED IN THE PROPOSED HCAPSNET FOR 25 × 25 WINDOW SIZE WITH PCS = 15 FOR UP & SV DATA PATCHES

Datasets	Layers	Shape	Output	Stride	BN	Padding	Activatin Function	Parameters
	Input Layer	-	(25, 25, 15, 1)	-	-	-	-	0
	Conv3D-1	(9, 9, 2)	(17, 17, 9, 8)	1	Yes	No	LeakyReLU	4544
	Conv3D-2	(9, 9, 2)	(9, 9, 5, 16)	1	No	No	LeakyReLU	51856
UP & SV	Conv2D	(3,3)	(7, 7, 64)	1	Yes	Yes	LeakyReLU	46144
	PrimaryCaps & reshape & squash	-	(392, 8)	-	-	-	LeakyReLU & Squash	0
	ClassCapsule	-	(9, 16)	-	-	-	-	451584
	Mask	-	(144)	-	-	-	-	0
	Decoder	-	(25, 25, 14, 1)	-	-	-	-	19875646
								Total trainable parameters: 20,429,774

TABLE III
DATASET STATISTICS AND DESCRIPTION

Datasets	Spatial Dimension (Pixels)	Spectral Dimension (Bands)	Wavelength Range (nm)	No. Labels	No. Classes	Location
Indian Pines (IP)	145 × 145	200	400 - 2500	10,249	16	Indian Pines, North-western Indiana
Salinas Valley (SV)	512 × 217	204	360 - 2500	5,4129	16	Salinas Valley, California California
University of Pavia (UP)	610 × 340	103	430 - 860	50,232	9	Pavia Northern Italy



No.	Color	Class Names	No. of Samples
1	🟧	Alfalfa	46
2	🟥	Corn-notill	1428
3	🟩	Corn-mintill	830
4	🟦	Grass-pasture	483
5	🟨	Grass-trees	730
6	🟦	Hay-windrowed	478
7	🟪	Soybean-notill	972
8	🟥	Soybean-mintill	2455
9	🟩	Soybean-clean	593
10	🟪	Woods	1265
11	🟦	Corn	237
12	🟪	Grass-pasture-mowed	28
13	🟫	Oats	20
14	🟩	Wheat	205
15	⬜	Buildings-Grass-Trees-Drives	386
16	🟥	Stone-Steel-Towers	93
Total number of samples: 10249			

Fig. 6. Natural composite images and labels for Indian Pines (IP).

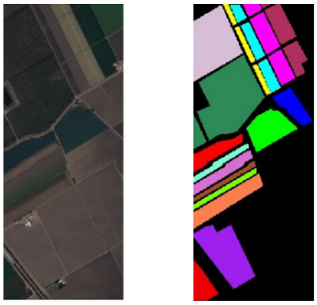
54 129, respectively. The total number of the samples in the UP dataset is 50 232 with 9 different classes. The natural composite images and labels for each dataset are illustrated in Figs. 6–8.

Although DL networks require a large number of training samples to achieve valid trained parameters and avoid overfitting, CapsNet-based models can be trained with limited training data. Therefore, to determine the performance of the approaches, we adjusted training samples to evaluate performance when 30%, 10%, 5%, and 1% are used for each dataset. In the training phase of HCapsNet, the transformation matrix, neuron weights, and biases are optimized.

F. Analysis Description

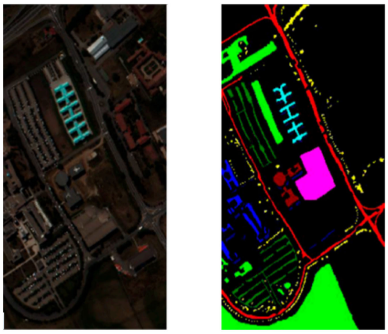
The representation of a data cube in HSI can be denoted by $C \in \mathbb{R}^{W \times H \times B}$ where W is the width, H is the height, and B is the depth so that, each pixel in C represents a vector with a length of B (the number of bands) from a set of different elements of the matrix $W \times H$. To elaborate, each of these vectors forms a particular element in the captured scene with individual spectral specifications and can be considered a high dimensional dataspace. We first split the input data into 3-D-patches $(w \times h \ \times b)$, with the label centered pixel at $[\frac{w}{2} + 1, \ \frac{h}{2} + 1]$.

In HSI processing, we are generally dealing with limited training samples. Therefore, to set up the hyperparameters to



No.	Color	Class Names	No. of Samples
1	🟥	Brocoli_green_weeds_1	1977
2	🟩	Brocoli_green_weeds_2	3726
3	🟦	Fallow	1976
4	🟨	Fallow_rough_plow	1394
5	🟦	Fallow_smooth	2678
6	🟪	Stubble	3959
7	🟥	Celery	3579
8	🟩	Grapes_untrained	11213
9	🟪	Soil_vinyard_develop	6197
10	🟧	Corn_senesced_green_weeds	3249
11	🟦	Lettuce_romaine_4wk	1058
12	🟪	Lettuce_romaine_5wk	1908
13	🟫	Lettuce_romaine_6wk	909
14	🟩	Lettuce_romaine_7wk	1061
15	⬜	Vinyard_untrained	7164
16	🟥	Vinyard_vertical_trellis	1737
Total number of samples: 53785			

Fig. 7. Natural composite images and labels for Salinas.



No.	Color	Class Names	No. of Samples
1	🟥	Asphalt	6631
2	🟩	Meadows	18649
3	🟦	Gravel	2099
4	🟨	Trees	3064
5	🟦	Painted metal sheets	1345
6	🟪	Bare Soil	5029
7	🟥	Bitumen	1330
8	🟩	Self-Blocking Bricks	3682
9	🟦	Shadows	947
Total number of samples: 42776			

Fig. 8. Natural composite images and labels for University of Pavia.

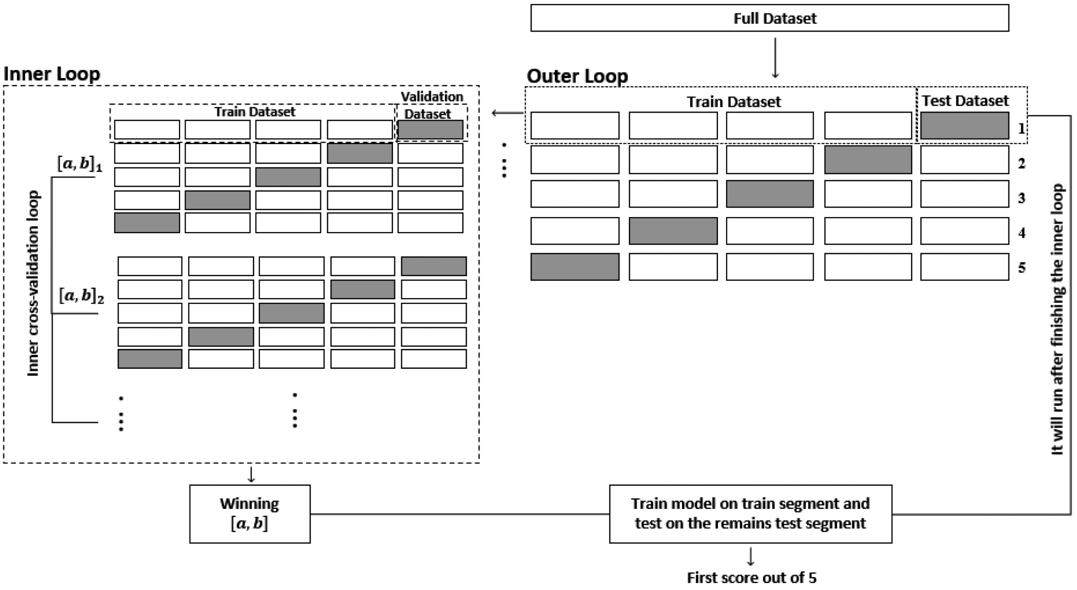


Fig. 9. Inner and outer loops operation in nested-CV. To begin, the data are divided into k parts (k-1 train datasets and 1 test dataset). The test dataset is kept in the outer-loop for testing. The remaining k-1 parts are then used to create an inner-loop. A validation accuracy is collected for each hyperparameter combination once the inner-loop data have been separated again into k parts. As a result of this, k values for the model validation accuracy are obtained for each parameter specified in the hyperparameter space. The ideal set of hyperparameters is determined by calculating the mean validation accuracy for each HP combination across all inner parts. Then based on the summation of all inner-loop validation accuracy, a single set of hyperparameters is chosen. The optimized hyperparameters are then applied to the final model, which is trained on outer-loop train parts and tested on outer-loop test parts, with the resulting classification accuracy employed to indicate model performance.

obtain higher accuracy and estimate the correct error unbiasedly, we utilized nested-CV. Although nested-CV is applied to models in case of a limited dataset, we cannot overlook the fact that it is computationally expensive in larger datasets [65]. However, it can be applied to model training where optimisation is required on hyperparameters. If nested-CV is not employed to choose the model, the related data will be used to fit the model parameters and evaluate the performance of the model, which may cause overfitting due to data leakage into the model [66]. The model stability and the size of the dataset are two factors that influence overfitting. Therefore, Nested-CVs involving a set of train, validation, and tests is applied to eliminate such an occurrence [67]. There are two loops in this algorithm. In outer and inner loops, data is split into five parts. Fig. 9 illustrates how the inner and outer loops operate for optimizing parameters set a and b. As can be seen, in the inner loop, by fitting the model to each training dataset, it attempts to maximise the accuracy. Then the hyperparameters are maximized by fitting the data to the validation set. Finally, the generalisation error estimation in the outer loop is obtained using the average outer loop test set scores which are reported in the results section and use to compare approaches.

TABLE IV
HYPERPARAMETERS SETTINGS USING NESTED-CV

HPs Methods	Parameter 1	Parameter 2
HCapsNet CAPSNET	$n: 6, 8, 10$	$m: 14, 16, 18$
HybridSN	# Spatial 2D filter: $(16, 32, 64)$	# Spectral- Spatial 3D filter: $(16, 32, 64)$
2D-CNN 3D-CNN	The first layer filters $(16, 32, 64)$	The last layer filters $(16, 32, 64)$
SVM	gamma: $(1, 5, 15, 20)$	C: $(0.0001, 0.001, ..., 1000)$
SCAE	kernels sizes: $(3, 4, 5)$	# Kernels: $(32, 64, 128)$
CRNN	Convolutional kernel sizes: $(3, 4, 5)$	# kernels for recurrent layers: $(16, 32, 64)$

The classification results for our HCapsNet method are compared with other classic classification methods available in literature, including spectral-based classifiers: SVM [12], spatial-based classifier: 2-D-CNN [56], spatial-spectral classifier: 3-D-CNN [57], HybridSN [34], SSRN [39], SCAE [31], CRNN [32], and CAPSNET [54] as the state-of-the-art methods. For all networks, we selected architectures derived from their original structure. For each method we optimized hyperparameters within the nested-CV as shown in Table IV. For HCapsNet to determine the best parameters, vector lengths n and m are selected between $(n: 6, 8, 10)$ and $(m: 14, 16, 18)$. The number of filters implemented in the 2-D-CNN and 3-D-CNN classifiers is chosen between 16, 32, 64. The same number of filters are examined for spatial and spectral-spatial filters in HybridSN. In our experiment, kernel filters for the SVM are searched for a range of gamma (1,5,15,20), C $(0.0001, 0.001, \ldots, 1000)$. SCAE employs three convolutional layers and three deconvolutional layers with kernels sizes selected between (3, 4, 5). The number of the kernels in the first, second, and third convolutional layers are set between (32, 64, and 128). In CRNN, two recurrent layers with convolutional LSTM units are used. For both recurrent layers, convolutional kernels are selected between $(3, 4, 5)$. The first and

TABLE V
CLASSIFICATION RESULTS FOR IP DATASET

Training Samples	Methods	SVM [56]	2D-CNN [56]	3D-CNN [57]	HybridSN [34]	SSRN [39]	SCAE [31]	CRNN [32]	CAPSNET [54]	HCapsNet
30%	OA (%)	90.30 ± 2.8	89.48 ± 0.2	91.10 ± 0.4	99.75 ± 0.1	99.18± 0.1	99.32± 0.3	99.89± 0.4	99.85± 0.1	99.86± 0.1
	AA (%)	91.03 ± 2.7	86.14 ± 0.8	91.58 ± 0.2	99.63 ± 0.2	98.94± 0.0	98.44± 0.2	99.58± 0.7	99.74± 0.1	99.92± 0.0
	Kappa (%)	92.10 ± 3.2	87.96 ± 0.5	89.98 ± 0.5	99.71 ± 0.1	99.02± 0.1	99.65± 0.1	99.45± 0.3	99.77± 0.1	99.04± 0.2
10%	OA (%)	88.22 ± 1.05	80.27± 0.2	82.62± 0.5	98.39± 0.6	98.40± 0.2	98.68± 0.3	96.78± 0.5	98.00± 0.2	98.55± 0.1
	AA (%)	91.40 ± 2.60	68.32± 0.3	76.51± 0.7	98.01± 0.6	86.14± 1.5	97.83± 0.8	94.47± 0.8	97.29± 0.1	98.37± 0.1
	Kappa (%)	86.86 ± 1.19	78.26± 0.5	76.25± 0.4	98.00± 0.3	98.33± 1.1	98.49± 0.1	96.33± 0.4	96.44± 0.1	97.35± 0.3
5%	OA (%)	86.07 ± 1.52	75.32± 0.2	75.85± 0.1	95.02± 0.7	95.33± 1.5	96.61± 0.5	94.15 ± 0.8	95.30± 0.0	97.34± 0.1
	AA (%)	88.07 ± 2.27	73.04± 0.6	73.90± 0.7	94.89± 0.4	93.54± 2.2	94.58± 1.5	90.30 ± 4.1	94.45± 0.1	96.57± 0.2
	Kappa (%)	84.46 ± 1.71	72.25± 0.5	73.74± 0.8	94.31± 0.1	95.42± 1.4	96.13± 0.1	93.50 ± 1.0	94.12± 0.2	96.24± 0.3
1%	OA (%)	81.01 ± 2.10	68.05± 0.1	70.69± 0.6	76.25± 0.1	85.18± 3.4	87.11± 1.4	85.25± 0.2	87.56± 0.1	90.67± 0.2
	AA (%)	80.79 ± 4.69	69.65± 0.5	71.99± 0.2	79.44± 0.4	80.02± 3.7	84.78± 0.5	82.33± 1.4	81.33± 0.1	89.21± 0.2
	Kappa (%)	78.77 ± 2.34	69.08± 0.1	71.08± 0.4	76.05± 0.5	85.60± 2.5	86.73± 0.1	82.13± 0.3	86.21± 0.1	89.12± 0.3

TABLE VI
CLASSIFICATION RESULTS FOR UP DATASET

Training Samples	Methods	SVM [56]	2D-CNN [56]	3D-CNN [57]	HybridSN [34]	SSRN [39]	SCAE [31]	CRNN [32]	CAPSNET [54]	HCapsNet
30%	OA (%)	94.34 ± 1.2	97.86 ± 0.2	96.53 ± 0.1	99.98 ± 0.0	99.89± 0.1	99.87± 0.5	99.10± 0.1	99.99± 0.0	99.99± 0.0
	AA (%)	92.98 ± 0.4	96.55 ± 0.0	97.57 ± 1.3	99.97 ± 0.0	99.90± 0.1	99.68± 0.1	99.20± 0.1	99.94± 0.0	99.98± 0.1
	Kappa (%)	92.50 ± 2.7	97.16 ± 0.5	95.51 ± 0.2	99.98 ± 0.0	99.88± 0.0	99.65± 0.2	98.77± 0.2	99.97± 0.1	99.98± 0.1
10%	OA (%)	91.04± 1.5	96.63± 0.2	96.34± 0.5	99.72± 0.6	99.62± 0.2	99.88± 0.6	96.00± 0.4	98.12± 0.3	98.54± 0.2
	AA (%)	90.64± 3.9	94.84± 0.3	97.03± 0.8	99.20± 0.1	99.49± 0.0	99.65± 0.1	97.18± 0.6	97.33± 0.2	97.74± 0.3
	Kappa (%)	91.77± 1.4	95.53± 0.2	94.90± 0.1	99.64± 0.6	99.50± 0.5	99.75± 0.1	95.18± 1.0	96.53± 0.1	96.94± 0.1
5%	OA (%)	87.23± 1.7	88.24± 0.4	86.22± 0.6	75.54± 0.8	98.22± 0.6	99.41± 0.1	97.11± 0.5	98.63± 0.0	96.75± 0.2
	AA (%)	88.88± 2.2	86.25± 0.6	85.45± 0.8	74.24± 0.6	97.59± 0.3	99.01± 0.3	98.27± 0.7	97.79± 0.1	95.82± 0.1
	Kappa (%)	86.99± 1.3	85.67± 0.6	83.24± 0.6	73.22± 0.8	98.43± 1.3	99.21± 0.1	96.09± 1.2	95.01± 0.1	95.91± 0.1
1%	OA (%)	70.03± 2.7	75.24± 0.7	78.01± 0.2	68.05± 0.8	88.78± 1.3	90.25± 0.2	90.42± 0.4	90.55± 0.0	95.57± 0.3
	AA (%)	82.73± 1.9	74.98± 0.1	77.22± 0.1	67.55± 0.7	82.08± 2.6	85.88± 0.1	87.50± 0.6	84.41± 0.1	91.39± 0.1
	Kappa (%)	70.70± 1.7	74.55± 0.5	78.74± 0.3	66.98± 0.5	87.70± 2.5	89.52± 0.2	89.89± 0.1	89.11± 0.0	94.09± 0.2

second recurrent layers' kernel numbers are set between (16, 32, 64). For all DL approaches the number of training epochs and the mini-batch sizes were fixed to 100 and 64, respectively. To enable a fair comparison for all methods the number of HSI cubes in the train and test datasets are consistent.

Also, PCA is used for dimensionality reduction and the optimal number of PCs are selected from the validation data ranging from 15–30 PCs. A network structure's performance can also be measured by the computation time, which directly measures its computational efficiency. The computational complexity of deep learning models is heavily influenced by network parameters. Therefore, we compared computational time of HCapsNet with other methods. Furthermore, an ablation study is performed to demonstrate the effectiveness of each part of HCapsNet. Since the reconstruction part drives the network to push all the necessary information to the top layer of the HCapsNet, it is important to compare HCapsNet with and without the reconstruction section (Decoder). Also, to show the efficacy of other components, HCapsNet with and without Conv2-D and Conv3-D is analyzed.

G. Statistics

To evaluate the performance of the model, we employ overall accuracy (OA) and average accuracy (AA) quantitative metrics. Specifically, OA represents the average correct classification item across all classes, while AA specified the number of correct classification samples per class for all test samples. Also, the Kappa statistic, the estimation for the agreement between labels of classified instances in the machine and ground truth labels, are reported.

Analysis of variance (ANOVA) is used to determine the significance of differences between the results of each method ($p < 0.05$). In addition, the Tukey method as an ANOVA post hoc analysis is used to compare the pairwise combinations of the methods.

III. RESULTS

This section presents the accuracy of methods, computational time, hyperparameter selection, and ablation study.

A. Accuracy of Methods

The classification results of all methods are presented in Table V–VII. Although there are no significant differences ($p < 0.05$) between the proposed method and state-of-the-art DL methods with larger training set sizes ($> 1\%$), the HCapsNet provided higher accuracies using the least training data.

For example, for the SV dataset and 1% training samples sizes, ANOVA revealed significant differences in performance across all methods ($F_{8,26} = 32.4$,p $< 1 \times 1 \times 10^{-10}$). A post hoc

TABLE VII
CLASSIFICATION RESULTS FOR SV DATASET

Training Samples	Methods	SVM [56]	2D-CNN [56]	3D-CNN [57]	HybridSN [34]	SSRN [39]	SCAE [31]	CRNN [32]	CAPSNET [54]	HCapsNet
30%	OA (%)	98.95 ± 0.3	97.38 ± 0.0	99.96 ± 0.2	99.99 ± 0.0	99.97± 0.1	99.82± 0.6	99.38± 1.1	99.81± 0.0	99.98± 0.0
	AA (%)	98.60 ± 0.3	98.84 ± 0.1	97.01 ± 0.6	99.99 ± 0.0	99.96± 0.1	99.75± 0.4	98.86± 0.4	99.92± 0.0	99.99± 0.0
	Kappa (%)	98.11 ± 0.2	97.08 ± 0.1	98.32 ± 0.5	99.99 ± 0.0	99.96± 0.0	99.83± 0.3	99.21± 0.9	99.79± 0.1	99.98± 0.0
10%	OA (%)	95.10± 0.1	92.67± 0.2	98.38± 0.0	99.88± 0.1	99.63± 0.1	99.75± 0.7	98.29± 0.9	99.96± 0.0	99.98± 0.0
	AA (%)	97.50± 0.2	92.06± 0.4	97.91± 0.1	99.88± 0.1	99.77± 0.2	99.71± 0.8	97.77± 1.3	99.97± 0.1	99.99± 0.0
	Kappa (%)	93.53± 0.3	91.15± 0.3	96.14± 0.1	99.88± 0.1	99.50± 0.3	99.73± 0.1	98.10± 1.0	99.95± 0.0	99.97± 0.0
5%	OA (%)	87.51± 0.8	86.12± 0.1	88.11± 0.2	95.36± 0.2	91.12± 1.7	97.98± 0.3	96.12 ± 0.4	95.02± 0.2	97.12± 0.1
	AA (%)	88.77± 1.2	87.32± 0.4	89.01± 0.5	94.21± 0.3	93.50± 2.5	98.41± 0.3	94.91 ± 0.8	94.40± 0.1	96.50± 0.2
	Kappa (%)	86.88± 1.4	85.50± 0.2	87.36± 0.2	93.22± 0.1	90.87± 1.2	97.76± 0.1	95.68 ± 0.5	95.00± 0.1	97.03± 0.3
1%	OA (%)	83.19± 1.3	80.33± 0.2	74.41± 0.3	77.25± 0.2	85.54± 3.6	85.18± 0.2	85.10± 0.3	86.01± 0.0	90.01± 0.1
	AA (%)	84.80± 1.9	81.45± 0.6	75.44± 0.4	76.98± 0.7	82.98± 2.3	80.98± 0.1	79.81± 0.7	83.80± 0.1	89.86± 0.3
	Kappa (%)	82.95± 0.9	80.70± 0.2	74.21± 0.1	76.90± 0.6	84.55± 3.1	84.02± 0.1	83.66± 0.2	85.60± 0.2	90.65± 0.2

TABLE VIII
NUMBER OF THE NETWORK PARAMETERS AND TIME CONSUMPTION

Methods	HCapsNet	CAPSNET	CRNN	SCAE	SSRN	HybridSN	3D-CNN	2D-CNN	SVM
No. of trainable Parameters for IP	20,919,246	10,864,846	76,864	88,254	54,769	5,122,176	553,152	155,568	-
Time consumption (s) Train:	139.9	103.2	58.3	60.6	41.1	89.6	39.3	56.5	208.3
Time consumption (s) Test:	6.9	5.1	2.9	3.3	2.0	4.4	1.9	2.8	1.4

analysis indicated that HCapsNet performed significantly better than all other approaches. Also, with 85.54% overall accuracy, SSRN outperformed all other CNN-based approaches, including CRNN, SCAE, HybridSN, 2-D-CNN, and 3-D-CNN but not HCapsNet.

Similarly, for IP and UP datasets, HCapsNet achieved 90.67% and 95.57% in comparison to the second-best method (CAPSNET) with 87.56% and 90.55% overall accuracies, respectively. The ANOVA test indicated significant differences for IP $(F_{8,26} = 39.5, p < 1 \times 1 \times 10^{-11})$ and UP $(F_{8,26} = 23.8, p < 1 \times 1 \times 10^{-9})$ with smaller (1%) sample sizes. For IP, CRNN and SCAE outperformed all other CNN-based methods with 85.25% and 87.11% (for UP 90.42% and 90.25%) overall accuracy. Although HCapsNet overall accuracy is significantly higher than SSRN and CRNN $(p < 0.05)$, the Tukey post hoc tests did not indicate a significant difference between HCapsNet, CAPSNET, and SCAE $(p > 0.05)$ for the IP dataset. Also, because CNN-based models may fail to generalise with fewer training samples, both 2-D and 3-D CNNs achieved 80.33% and 74.41%, respectively, whereas the SVM model performed better with 83.19% overall accuracy.

B. Computational Time

The computational efficiency of HCapsNet compared to other models are shown in Table VIII. HCapsNet's training times are approximately 40(s) longer than its counterpart (CAPSNET), owing to the fact that both 2-D and 3-D CNNs are used in HCapsNet. CAPSNET, on the other hand, employs only 2-D CNNs. Furthermore, when comparing HCapsNet to HybridSN, which used similar 2-D and 3-D CNNs, HCapsnet acquired more trainable parameters because of the use of dynamic routing in HCapsNet. In other words, the dynamic routing method in the HCapsNet demands a significantly greater amount of computational resources than its CNN counterparts.

TABLE IX
HYPERPARAMETERS SELECTED USING NESTED-CV

HPs / Methods	Parameter 1	Parameter 2
HCapsNet, CAPSNET	n : 8	m : 16
HybridSN	2D filter: 16	3D filter: 32
2D-CNN , 3D-CNN	1st layer filters: 64	1st layer filters: 32
SVM	gamma: 5	C: 0.001
SCAE	kernels sizes: 3	# kernels: 64, 128
CRNN	kernel sizes:3	#1st and 2nd kernels: 32, 64

C. Hyperparameter Selection

The hyperparameters selected for methods are reported in Table IX. In the case of the HCapsNet, optimal vector lengths were selected as $n = 8$ and $m = 16$ using nested-CV. These are the optimal length for one single primary capsule and one ClassCapsule.

In addition, the impact of employing a different number of PCA on datasets was analyzed. The classification results for HCapsNet for 15, 20, 25, and 30 PCs reported in such a way that all the framework settings were retained as before. It can be seen in Fig. 10 that the classification results on the SV dataset did not increase significantly $(p > 0.05)$ due to using more information from the data. Therefore, the DR is not sensitive to the number of PCs, and the computational cost in DR will be relatively

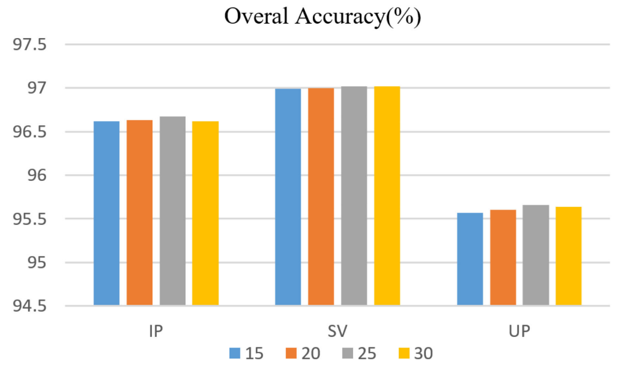


Fig. 10. Effect of different principal components (PCs) on different datasets.

TABLE X
IMPACT OF DIFFERENT PATCHING WINDOW SIZES AND CNN KERNEL SIZES

Window Size / Dataset	19	23	25	27
OA (%) for IP	90.01±0.4	90.23±0.3	**90.67±0.2**	89.33±0.4
OA (%) for UP	**95.57±0.3**	95.01±0.1	95.12±0.1	94.50±0.1
OA (%) for SA	89.25±0.2	90.00±0.1	**90.01±0.1**	89.65±0.2

increased by using a higher number of PCs without additional accuracy gains.

Since the patches' windows sizes and convolution filter sizes play a pivotal role in the HCapsNetarchitecture, as they determine the number of the capsules in the primary capsule layer, these parameters are also examined in the presented structure. For this, we fixed the number of PCs to 25 for the IP dataset and 15 for SA and UP datasets, respectively. In order to analyze the window sizes, the size of 3-D-Conv filters is set to $(9, 9, 7)$ and $(9, 9, 5)$, and $(3, 3)$ for the 2-D-Conv. The classification results are reported in Table X, while only 1% data are used for training. As can be seen in Table X, for the best performance of the network 25 is the best window size for IP and SA, and 19 for UP.

The quality of the classification results shows the effectiveness of the different classification methods. The classification maps for the different trained models after optimizing parameters are shown in Figs. 11–13. The ground truth and classification maps comparison show that the HCapsNet architecture with dynamic routing has better performance than the other spectral-spatial methods. Results also indicated that HybridSN performed significantly better than 2-D and 3-D CNNs due to use of spatial-spectral feature extraction in their structure. Also, in the boundaries of the classes in the 2-D-CNN, some artefacts can be observed which suggest the spatial information is not sufficient for classification in boundaries. Since only spectral information is employed in classical methods such as SVM, a number of noise issues are obvious. Therefore, a reasonable approach can be found in models using both spatial-spectral information with more consistency of the classes. HCapsNet achieved higher overall accuracy as it missed fewer pixels than other methods in the entire image, including boundaries.

D. Ablation Study

To better demonstrate the effectiveness of each part of HCapsNet, the performance of the model without the reconstruction part (Decoder), Conv2-D, and Conv3-D are analyzed. The experimental results on datasets are summarized in Table XI. When compared to other methods, HCapsNet using a Decoder yield higher classification accuracies ($F_{3,11} = 7.9$, $p < 1 \times 1 \times 10^{-3}$). As a result, the number of trainable parameters and the time consumption is increased. Since there was no decoder used in without-decoder, the margin loss has been used instead of total loss. To determine the significance of CNNs in our proposed architecture, we compared the effects of each 2-D and 3-D CNN. HCapsNet (With-decoder) achieved a significantly higher overall accuracy than Without 2-D-CNN and 3-D-CNNs ($p < 0.02, p < 0.01$). These results indicate that efficient spectral-spatial feature extraction is critical for enhancing HCapsNet classification performance. There are insignificant differences in performance when comparing HCapsNet Without 2-D-CNN to HCapsNet without 3-D-CNNs ($p > 0.05$) indicating that advantages of the hybrid approach is realized with combined 2-D and 3-D CNN decoders.

IV. DISCUSSION

Here, we presented the HCapsNet and thoroughly evaluated its performance on three commonly used HSI benchmarks. The results supported the assertion that DL methods as a data-managed features technique and, particularly, HCapsNet, are incredibly performant in the situation of small training sample sizes. In terms of encoding the characteristics and feature spatial relationships, the proposed architecture is a novel and unique method in HSI processing. HCapsNet dealt with the challenging issue of natural complexity in HSI, which is caused by the high spectral resolution. To elaborate, in the proposed model, management of the spatial-spectral features is possible with the extraction of more relevant information in the hierarchy of part of the image.

In addition, as can be seen in the results for all three datasets, CapsNet-based and CNN-based models performed better than SVM—a manually engineered features technique. Although the result for SVM showed that the accuracy diminished considerably when smaller training samples were used, the HCapsNet maintained stable accuracy. Results for HCapsNet and CAPSNET also indicated less uncertainty arising from performance variability in comparison to the other approaches compared as evident from lower and less variable standard deviation. Furthermore, the classification maps displayed more misclassification for SVM, 2-D-CNN, and 3-D-CNN, especially in the boundary regions, which is not the case for HCapsNet. This weakness is more evident for the UP dataset because this dataset has more complicated class borders than the others.

Since we deal with limited datasets in remote sensing [68], the most suitable parameters are achieved by using nested-CV which helps to prevent the leakage of the data into the model, but it does require much more computational effort. Therefore, in order to overcome the computational overhead and represent the input patches properly, we chose the fixed

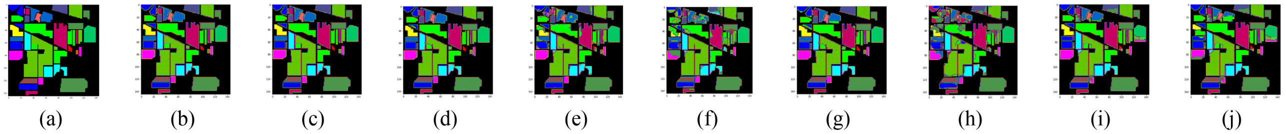
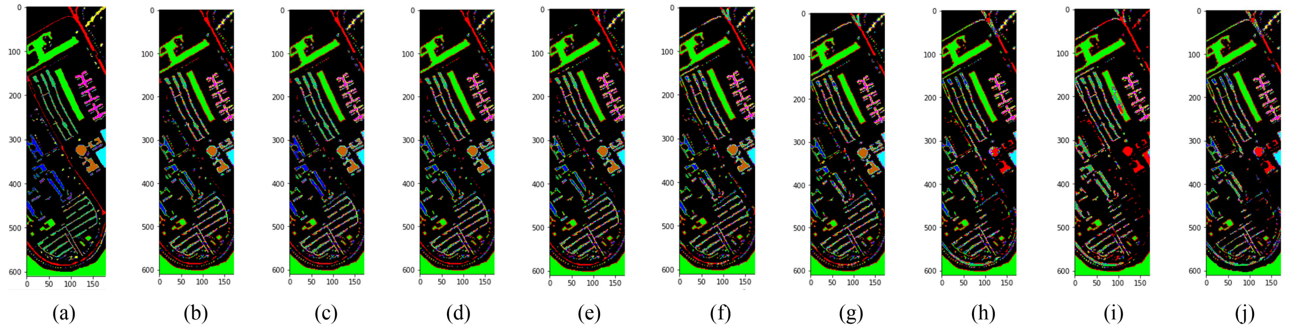
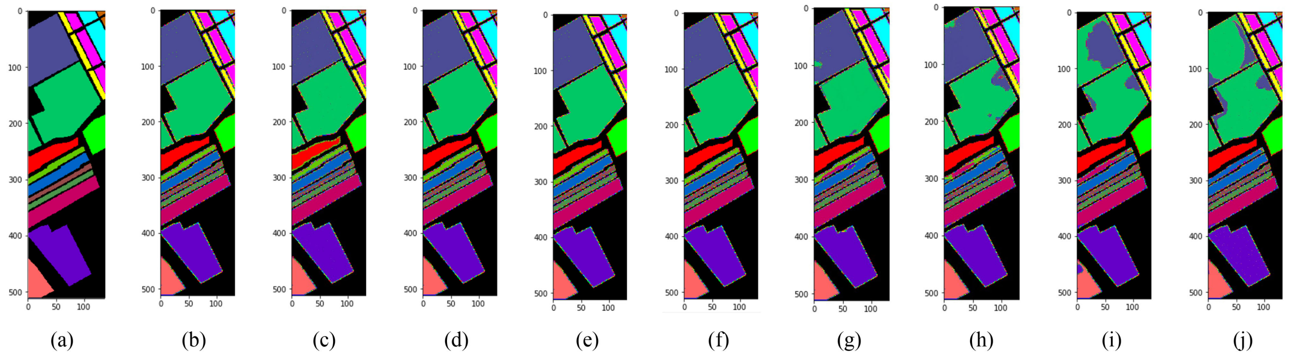

(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Fig. 11. Classification maps for IP dataset: Ground truth (a), the predicted output for HCapsNet, CAPSNET, CRNN, SCAE, SSRN, HybridSN, 3-D-CNN, 2-D-CNN, and SVM (b)–(j).



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Fig. 12. Classification maps for UP dataset: Ground truth (a), the predicted output for HCapsNet, CAPSNET, CRNN, SCAE, SSRN, HybridSN, 3-D-CNN, 2-D-CNN, and SVM (b)–(j).



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Fig. 13. Classification maps for SV dataset: Ground truth (a), the predicted output for HCapsNet, CAPSNET, CRNN, SCAE, SSRN, HybridSN, 3-D-CNN, 2-D-CNN, and SVM (b)–(j).

TABLE XI
Ablation Study on Components of HCapsNet: the Performance of the Different Designs, Loss, the Number of Trainable Parameters, and Computational Time

Factors Models	IP (OA%)	UP (OA%)	SV (OA%)	Loss	No. of trainable Parameters	Computational Time (s)
HCapsNet	90.67± 0.2	95.57± 0.3	90.01± 0.1	Total Loss	UP , SV: 20,429,774 IP: 20,919,246	139.9 153.3
HCapsNet Without-Decoder	84.39± 0.8	87.45± 0.6	82.33± 0.5	Margin Loss	UP , SV: 554,128 IP: 1,043,600	86.1 98.1
HCapsNet Without 2D-CNN	83.23± 0.2	86.52± 0.1	82.33± 0.1	Total Loss	UP , SV: 20,383,630 IP: 20,734,862	128.5 150.2
HCapsNet Without 3D-CNNs	82.56± 0.1	85.55± 0.0	81.01± 0.0	Total Loss	UP , SV: 20,373,374 IP: 20,864,846	127.5 150.3

patch sizes of 25. This also helped us to have a fair comparison with other methods. In addition, the learning process is stabilized by BN as a regularization method in the model and can result in higher accuracy [69]. As suggested in the literature the higher number of labeled input cubes will improve the performance of the models, in order to compare different models, the same size of the training samples should be considered [70]. Regarding this, we believe that with an equal standard of comparison, the HCapsNet can outperform other models.

In terms of computation time, CapsNet-based architectures are more time consuming than other approaches. The CapsNet-based models require approximately double the time to train compared to the CNN-based models, indicating that the CapsNet is more computationally expensive. This is because their architecture incorporates an iterative dynamic routing. When we are dealing with limited training samples, the parameter number in CapsNet-based architectures is much higher than CNN-based models; however, adopting a deeper CNN-based model with more trainable parameters may cause overfitting, which is not the case the CapNet approaches.

Also, as shown in the ablation study, HCapsNet with a decoder achieves higher classification accuracy; as a result, the number of trainable parameters and the amount of time required to train the network increases. Comparing the results for different parts of the HCapsNet confirmed that efficient spectral-spatial feature extraction is crucial for improving the classification performance of the HCapsNet.

The architecture currently used in HCapsNet is relatively strict. In this structure, a vector representation of the instantiation parameters is applied and there is only one capsule layer (primary caps) before the final ClassCapsule layer. Therefore, for generalisation and applying the raw HSI format in case of limited training sample sizes, the latest version of the CapsNet referred to as matrix capsule, which transforms the vector formation to matrix formation of the capsules, may be considered [71].

In addition, there are some restraints in the design of neural networks or learning frameworks. For example, unlimited design choices and constrained receptive fields to particular contexts due to geometric restrictions with CNN kernels. Although numerous studies have examined practical limits in the remote sensing community for regularizing HSI datasets in diverse applications such as spectral unmixing, sampling strategy [72], [73], constraints on architecture spaces to avoid prohibitively expensive neural architecture search has not been addressed. Recently transformer neural networks [74] which use the attention mechanism have sparked interest across a wide variety of problems, including HSI. For example, Zhong *et al.* [75] presented a framework for architecture search that combines neural architecture search and experts' knowledge to alleviate the computational cost of architecture search. Therefore, to generalize spectral-spatial features and reduce the computational cost, which was the main drawback in our method, an architecture search framework in HCapsNet, should be developed and evaluated in future research.

V. Conclusion

In this article, a review of the fundamental CapsNet with the DR algorithm was presented. In addition, an end-to-end DL architecture (HCapsNet) for HSI classification in situations with limited training data was proposed so that the combination of 2-D and 3-D CNNs was employed in the feature extractor. Generally, DL methods in HSI rely on CNNs to extract spatial and spectral features. However, since CNNs are often used in conjunction with pooling, they cause the loss of valuable information in the image. In addition to mistreating information, pooling also eliminates the hierarchy of parts. In HCapsNet, the instantiation parameters which form a vector are determined by estimating the probability of the presence of the spatial-spectral features in the HSI data cube. These features are preserved efficiently in DR. Furthermore since the hyperparameter tuning is a challenging issue with respect to limited training data, nested-CV was applied to prevent data leakage into the models and ensure generalization capability is thoroughly evaluated. The significantly better performance of HCapsNet using limited training data (e.g., 1% of entire data) is demonstrated on three widely used hyperspectral image datasets. Smaller training sample size impacted the classification results and raised the risk of overfitting in the non-CapsNet models such as 2-D-CNN, 3-D-CNN, and HybridSN; however, the consistency in the classification results provided by HCapsNet architecture regardless of the amount of training data is evidence of the potential for the proposed HCapsNet architecture.

References

[1] L. Fasnacht, M.-L. Vogt, P. Renard, and P. Brunner, "A 2D hyperspectral library of mineral reflectance, from 900 to 2500 nm," *Sci. Data*, vol. 6, no. 1, pp. 1–7, 2019.

[2] C. Chion, J.-A. Landry, and L. Da Costa, "A genetic-programming-based method for hyperspectral data information extraction: Agricultural applications," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 8, pp. 2446–2457, Aug. 2008.

[3] U. Amato *et al.*, "Statistical classification for assessing PRISMA hyperspectral potential for agricultural land use," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 6, no. 2, pp. 615–625, Apr. 2013.

[4] S. K. Meerdink, D. A. Roberts, K. L. Roth, J. Y. King, P. D. Gader, and A. Koltunov, "Classifying California plant species temporally using airborne hyperspectral imagery," *Remote Sens. Environ.*, vol. 232, 2019, Art. no. 111308.

[5] H. van Deventer, M. A. Cho, O. Mutanga, L. Naidoo, and N. Dudeni-Tlhone, "Reducing leaf-level hyperspectral data to 22 components of biochemical and biophysical bands optimizes tree species discrimination," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 8, no. 6, pp. 3161–3171, Jun. 2015.

[6] P. Levin *et al.*, "A wafer level packaged fully integrated tunable Fabry-Pérot filter with extended optical range for multispectral and hyperspectral imaging," *J. Microelectromech. Syst.*, vol. 29, no. 3, pp. 357–369, 2020.

[7] G. M. Wyller *et al.*, "Correlation of defect luminescence and recombination in multicrystalline silicon," *IEEE J. Photovolt.*, vol. 9, no. 1, pp. 55–63, Jan. 2019.

[8] I. C. C. Acosta, M. Khodadadzadeh, L. Tusa, P. Ghamisi, and R. Gloaguen, "A machine learning framework for drill-core mineral mapping using hyperspectral and high-resolution mineralogical data fusion," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 12, no. 12, pp. 4829–4842, Dec. 2019.

[9] D. Krupnik and S. Khan, "Close-range, ground-based hyperspectral imaging for mining applications at various scales: Review and case studies," *Earth- Sci. Rev.*, vol. 198, 2019, Art. no. 102952.

[10] A. Wang, J. Lu, J. Cai, G. Wang, and T.-J. Cham, "Unsupervised joint feature learning and encoding for RGB-D scene labeling," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 4459–4473, Nov. 2015.

[11] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. Nasrabadi, and J. Chanussot, "Hyperspectral remote sensing data analysis and future challenges," *IEEE Geosci. Remote Sens. Mag.*, vol. 1, no. 2, pp. 6–36, Jun. 2013.

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

[13] J. Li, J. M. Bioucas-Dias, and A. Plaza, "Semisupervised hyperspectral image segmentation using multinomial logistic regression with active learning," *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 11, pp. 4085–4098, Nov. 2010.

[14] Y. Y. Tang, Y. Lu, and H. Yuan, "Hyperspectral image classification based on three-dimensional scattering wavelet transform," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 5, pp. 2467–2480, May 2015.

[15] T. V. Bandos, L. Bruzzone, and G. Camps-Valls, "Classification of hyperspectral images with regularized linear discriminant analysis," *IEEE Trans. Geosci. Remote Sens.*, vol. 47, no. 3, pp. 862–873, Mar. 2009.

[16] M. Khodadadzadeh, J. Li, A. Plaza, H. Ghassemian, J. M. Bioucas-Dias, and X. Li, "Spectral-spatial classification of hyperspectral data using local and global probabilities for mixed pixel characterization," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 10, pp. 6298–6314, Oct. 2014.

[17] B. Liu, X. Yu, P. Zhang, A. Yu, Q. Fu, and X. Wei, "Supervised deep feature extraction for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 4, pp. 1909–1921, Apr. 2018.

[18] J. Xia, P. Ghamisi, N. Yokoya, and A. Iwasaki, "Random forest ensembles and extended multiextinction profiles for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 1, pp. 202–216, Jan. 2018.

[19] S. Li, W. Song, L. Fang, Y. Chen, P. Ghamisi, and J. A. Benediktsson, "Deep learning for hyperspectral image classification: An overview," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 9, pp. 6690–6709, Sep. 2019.

[20] G. Cheng, X. Xie, J. Han, L. Guo, and G.-S. Xia, "Remote sensing image scene classification meets deep learning: Challenges, methods, benchmarks, and opportunities," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 3735–3756, Jun. 2020.

[21] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks*," Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[22] A. R. Kosiorek, S. Sabour, Y. W. Teh, and G. E. Hinton, "Stacked capsule autoencoders," *Adv. Neural Inf. Process. Syst.*, vol. 32, pp. 15512–15522, 2019.

[23] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, no. 12, pp. 3371–3408 2010.

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

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

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[27] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1–9.

[28] G. Hinton *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.

[29] J. Li, X. Zhao, Y. Li, Q. Du, B. Xi, and J. Hu, "Classification of hyperspectral imagery using a new fully convolutional neural network," *IEEE Geosci. Remote Sens. Lett.*, vol. 15, no. 2, pp. 292–296, Feb. 2018.

[30] B. Rasti *et al.*, "Feature extraction for hyperspectral imagery: The evolution from shallow to deep: Overview and toolbox," *IEEE Geosci. Remote Sens. Mag.*, vol. 8, no. 4, pp. 60–88, Dec. 2020.

[31] R. Kemker and C. Kanan, "Self-taught feature learning for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 5, pp. 2693–2705, May 2017.

[32] Q. Liu, F. Zhou, R. Hang, and X. Yuan, "Bidirectional-convolutional LSTM based spectral-spatial feature learning for hyperspectral image classification," *Remote Sens.*, vol. 9, no. 12, 2017, Art. no. 1330.

[33] H. Guo, J. Liu, J. Yang, Z. Xiao, and Z. Wu, "Deep collaborative attention network for hyperspectral image classification by combining 2-D CNN and 3-D CNN," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 4789–4802, Aug. 2020.

[34] S. K. Roy, G. Krishna, S. R. Dubey, and B. B. Chaudhuri, "Hybridsn: Exploring 3-D-2-D CNN feature hierarchy for hyperspectral image classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 17, no. 2, pp. 277–281, Feb. 2020.

[35] D. Hong *et al.*, "Interpretable hyperspectral artificial intelligence: When nonconvex modeling meets hyperspectral remote sensing," *IEEE Geosci. Remote Sens. Mag.*, vol. 9, no. 2, pp. 52–87, Jun. 2021.

[36] D. Hong *et al.*, "More diverse means better: Multimodal deep learning meets remote-sensing imagery classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 5, pp. 4340–4354, May 2021.

[37] D. Hong, L. Gao, J. Yao, B. Zhang, A. Plaza, and J. Chanussot, "Graph convolutional networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 7, pp. 5966–5978, Jul. 2021.

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

[39] Z. Zhong, J. Li, Z. Luo, and M. Chapman, "Spectral-spatial residual network for hyperspectral image classification: A 3-D deep learning framework," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 2, pp. 847–858, Feb. 2018.

[40] M. Ye and Y. Guo, "Zero-shot classification with discriminative semantic representation learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 7140–7148.

[41] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 1199–1208.

[42] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 3856–3866.

[43] H. Chao, L. Dong, Y. Liu, and B. Lu, "Emotion recognition from multiband EEG signals using capsnet," *Sensors*, vol. 19, no. 9, 2019, Art. no. 2212.

[44] P. Afshar, A. Mohammadi, and K. N. Plataniotis, "Brain tumor type classification via capsule networks," in *Proc. 25th IEEE Int. Conf. Image Process.*, 2018, pp. 3129–3133.

[45] M. L. Mekhalfi, M. B. Bejiga, D. Soresina, F. Melgani, and B. Demir, "Capsule networks for object detection in UAV imagery," *Remote Sens.*, vol. 11, no. 14, 2019, Art. no. 1694.

[46] K. Suri and R. Gupta, "Continuous sign language recognition from wearable IMUs using deep capsule networks and game theory," *Comput. Elect. Eng.*, vol. 78, pp. 493–503, 2019.

[47] S. K. Roy, S. Manna, T. Song, and L. Bruzzone, "Attention-based adaptive spectral-spatial kernel resnet for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 9, pp. 7831–7843, Sep. 2021.

[48] 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, 2018, Art. no. 3153.

[49] W. Zhang, P. Tang, and L. Zhao, "Remote sensing image scene classification using CNN-capsNet," *Remote Sens.*, vol. 11, no. 5, 2019, Art. no. 494.

[50] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

[51] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 4278–4284.

[52] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learn. Represent*, Sep. 2014.

[53] K. Zhu, Y. Chen, P. Ghamisi, X. Jia, and J. A. Benediktsson, "Deep convolutional capsule network for hyperspectral image spectral and spectral-spatial classification," *Remote Sens.*, vol. 11, no. 3, 2019, Art. no. 223.

[54] M. E. Paoletti *et al.*, "Capsule networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 4, pp. 2145–2160, Apr. 2019.

[55] G. C. Cawley and N. L. Talbot, "On over-fitting in model selection and subsequent selection bias in performance evaluation," *J. Mach. Learn. Res.*, vol. 11, pp. 2079–2107, 2010.

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

[57] A. B. Hamida *et al.*, "Deep learning for semantic segmentation of remote sensing images with rich spectral content," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, 2017, pp. 2569–2572.

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

[59] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 221–231, Jan. 2013.

[60] Y. LeCun, C. Cortes, and C. Burges, "MNIST handwritten digit database," 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/.

[61] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[62] G. de Inteligencia Computacional (GIC), "Hyperspectral remote sensing scenes," 2020. [Online]. Available: http://www.ehu.eus/ccwintco/index.php/Hyperspectral-Remote-Sensing-Scen%es

[63] M. D. Farrell and R. M. Mersereau, “On the impact of PCA dimension reduction for hyperspectral detection of difficult targets,” *IEEE Geosci. Remote Sens. Lett.*, vol. 2, no. 2, pp. 192–195, Apr. 2005.

[64] B. Guo, S. R. Gunn, R. I. Damper, and J. D. Nelson, “Band selection for hyperspectral image classification using mutual information,” *IEEE Geosci. Remote Sens. Lett.*, vol. 3, no. 4, pp. 522–526, Oct. 2006.

[65] H. Van Hasselt, “Estimating the maximum expected value: An analysis of (nested) cross validation and the maximum sample average,” 2013, *arXiv:1302.7175*.

[66] R. Kohavi *et al.*, “A study of cross-validation and bootstrap for accuracy estimation and model selection,” in *Proc. Int. Joint Conf. Artif. Intell.*, 1995, pp. 1137–1145.

[67] J. Wainer and G. Cawley, “Nested cross-validation when selecting classifiers is overzealous for most practical applications,” *Expert Syst. Appl.*, vol. 182, p. 115222, Nov. 2018.

[68] K. Li, G. Wan, G. Cheng, L. Meng, and J. Han, “Object detection in optical remote sensing images: A survey and a new benchmark,” *ISPRS J. Photogrammetry Remote Sens.*, vol. 159, pp. 296–307, 2020.

[69] Z. Zhong, J. Li, L. Ma, H. Jiang, and H. Zhao, “Deep residual networks for hyperspectral image classification,” in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, 2017, pp. 1824–1827.

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

[71] G. E. Hinton, S. Sabour, and N. Frosst, “Matrix capsules with EM routing,” in *Proc. Int. Conf. Learn. Representations*, 2018.

[72] Z. Zhong, J. Li, D. A. Clausi, and A. Wong, “Generative adversarial networks and conditional random fields for hyperspectral image classification,” *IEEE Trans. Cybern.*, vol. 50, no. 7, pp. 3318–3329, Jul. 2020.

[73] D. Hong, N. Yokoya, J. Chanussot, and X. X. Zhu, “An augmented linear mixing model to address spectral variability for hyperspectral unmixing,” *IEEE Trans. Image Process.*, vol. 28, no. 4, pp. 1923–1938, Apr. 2019.

[74] A. Vaswani *et al.*, “Attention is all you need,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[75] Z. Zhong, Y. Li, L. Ma, J. Li, and W.-S. Zheng, “Spectral-spatial transformer network for hyperspectral image classification: A factorized architecture search framework,” *IEEE Trans. Geosci. Remote Sens.*, to be published, doi: 10.1109/TGRS.2021.3115699.

Massoud Khodadadzadeh received the B.S. degree in electrical engineering from the Sadjad University of Technology, Mashhad, Iran, in 2010 and the M.Sc. degree in electrical engineering from the Shahrood University of Technology, Shahrood, Iran, in 2014. He is currently working toward the Ph.D. degree in computer science with Intelligent Systems Research Centre (ISRC), School of Computing, Engineering and Intelligent Systems, Ulster University, Derry/Londonderry, U.K.

His research interests include advanced deep learning techniques, especially capsule neural network (CapsNet) method and its application on hyperspectral imaging, EEG, and coordination dynamics.

Xuemei Ding received the Ph.D. degree in computer science from Ulster University, Coleraine, U.K., in 2013.

She is currently a Lecturer in computer science with Cognitive Analytics Research Lab, Intelligent Systems Research Centre, School of Computing, Engineering and Intelligent Systems, Ulster University, Coleraine, U.K. Her research interests include machine learning, pattern recognition, artificial intelligence, novelty detection, and data science especially in relation to data analytics and predictive modeling on health care data such as Alzheimer’s disease and breast cancer.

Dr. Ding is a Fellow of The Higher Education Academy U.K. and a reviewer of some top journals and conferences.

Priyanka Chaurasia (Member, IEEE) received the B.Tech. degree in information technology from Harcourt Butler Technical University, Kanpur, India, in 2006 and the Ph.D. degree in computing and information engineering from Ulster University, Coleraine, U.K., in 2013.

Currently, she is a Lecturer in data analytics with the School of Computing & Intelligent Systems, Ulster University, Londonderry, U.K. Before her Ph.D., she was with IBM India Software Labs, Bangalore, India, for three years as a Software Engineer. She has patents granted by the U.S. Patent and Trademark Office in the area of signature verification. Her research interests include assistive technology, health care, activity recognition, biometric security, and data analytics.

Dr. Chaurasia was the recipient of the IBM Invention Achievement Award in 2008.

Damien Coyle (Senior Member, IEEE) received a first class degree in computing and electronic engineering and the Ph.D. degree in intelligent systems engineering from the University of Ulster, Derry, U.K., in 2002 and 2006, respectively. He is a Professor of Neurotechnology and the Director of the Intelligent Systems Research Centre and Research Director with the School of Computing, Engineering and Intelligent Systems, Ulster University, Coleraine, U.K. He has authored or coauthored over 160 research papers in areas such as computational intelligence/AI, bio-signal processing, computational neuroscience, neuroimaging, neurotechnology, and brain-computer interface (BCI) applications.

Dr. Coyle has won a number of prestigious international awards for his research including the 2008 IEEE Computational Intelligence Society (CIS) Outstanding Doctoral Dissertation Award and the 2011 International Neural Network Society (INNS) Young Investigator of the Year Award. He was an Ulster University Distinguished Research Fellow in 2011, a Royal Academy of Engineering/The Leverhulme Trust Senior Research Fellow in 2013, and a Royal Academy of Engineering Enterprise Fellow during 2016–2017. He is a UKRI Turing AI Fellow 2021–2025. He is a founding member of the International Brain-Computer Interface Society, chairs the IEEE Computational Intelligence Society (CIS) UK Ireland chapter, is the IEEE CIS representative and member on the steering committee of the IEEE Brain Technical Community and U.K. KTN Neurotechnology Innovation Network advisory board member. He is the Ulster lead of the Spatial Computing and Neurotechnology Innovation Hub (SCANi-hub) and the Northern Ireland High Performance Computing Facility (NIHPC) and co-investigator in Northern Ireland Functional Brain Mapping Facility (NIFBM), and leads a number of industry led data analytics projects via Ulster’s Cognitive Analytics Research Laboratory (CARL).