DEEP FEATURE EXTRACTION BASED ON DYNAMIC GRAPH CONVOLUTIONAL NETWORKS FOR ACCELERATED HYPERSPECTRAL IMAGE CLASSIFICATION

Quanwei Liu[1], Yanni Dong[1,*]
1 Hubei Subsurface Multi–scale Imaging Key Laboratory, Institute of Geophysics and Geomatics, China University of Geosciences, Wuhan 430074, China; liuquanwei@cug.edu.cn (Q.L.); dongyanni@cug.edu.cn (Y.D.)

Commission III, WG III/4

KEY WORDS: Hyperspectral images classification, Deep learning, Dynamic graph convolutional networks, Feature fusion, Regularization.

ABSTRACT:
Deep learning has achieved impressive results on hyperspectral images (HSIs) classification. Among them, supervised learning convolutional neural networks (CNNs) and semi-supervised learning graph neural networks (GNNs) are the two main network frameworks. However, 1) the supervised learning CNN faces the problem of high model time complexity as the number of network layers deepens; 2) the semi-supervised learning GNN faces the problem of high spatial complexity due to the computation of adjacency relations. In this paper, a novel dynamic graph convolutional HSI classification method is proposed, which is called dynamic graph convolutional networks (DGCNet). We first obtain two classification features by implementing flattening and global average pooling operation on the results of the convolution layer, which fully exploits the spatial-spectral information contained in the hyperspectral data. Then the dynamic graph convolution module is applied to extract the intrinsic structural information of each patch. Finally, HSI is classified based on spatial, spectral and structural features. DGCNet uses three branches to process multiple features of HSI in parallel and is trained in a supervised learning manner. In addition, DropBlock and label smoothing regularization techniques are applied to further improve the generalization capability of the model. Comparative experiments show that our proposed algorithm is comparable with the state-of-the art supervised learning models in terms of accuracy while also significantly outperforming in terms of time.

1. INTRODUCTION

Hyperspectral imaging technology is capable of providing detailed spectral information by sampling hundreds of narrow continuous spectral bands from the visible region (0.4-0.7 μm) to the short-wave infrared (SWIR) region (0.7-2.4 μm) (Ahmad et al., 2021). Applications based on hyperspectral images (HSIs) are widely developed, such as agricultural assessment, environmental inspection, et al. (Mahesh et al., 2015; Sabbah, 2012). However, the high-dimensional property of HSIs makes the topological relationship between high-dimensional units change essentially, such as the data distribution density becomes sparse and the spatial centroid is outside the hypersphere (Jimenez and Landgrebe, 1998). This makes the usual data processing methods based on color images cannot be directly applied to HSI to achieve the desired performance (Rasti et al., 2020). Therefore, a series of processing algorithms, such as target detection, change detection, and unmixing methods have been dedicated to develop for HSIs. Among them, the most basic one is HSI classification (HSIC) (Chang, 2021; Chen et al., 2016; Liu et al., 2019; Zhang et al., 2018).

The complex high-dimensional features of hyperspectral images are a challenge for traditional manual feature extraction and classification methods. After the development in the past few years, a series of DL models have been developed for HSIC. For instance, Chen (Chen et al., 2014) was the first to explore the SAE framework for hyperspectral classification. In his work, however, SAE can only extract higher-level features from one-dimensional data, while ignoring the two-dimensional spatial-spectral binding features. Unlike SAE, CNN uses fixed-size image patches for deep feature extraction (Liang and Li, 2016). Therefore, it can maintain the integrity of spatial information. In the work of Li et al. (Chen et al., 2016), wherein a spatial-spectral joint 3D-CNN HSIC network, which adopts an end-to-end training model, i.e., it no longer relies on complex data processing. Similarly, (Li et al., 2017) further investigated 3D CNNs for spatial-spectral classification using input squares of HSIs with smaller image patches. But the classification accuracy of the models decreases as the network becomes deeper.

However, on the one hand, the above methods usually have a high computational complexity and require a lot of time for training (Wang et al., 2018). On the other hand, limited labeled samples are a common issue in the practice of HSI classification, and GCN-based semi-supervised learning was found to be a good solution (He et al., 2021). By using GNNs, it is possible to mine the intrinsic connections between different vertexes and make full use of the rich spatial and spectral information of HSI (Kipf and Welling, 2017). For example, in (Dong et al., 2022), a graph attention network, which is a modification of GCN, was proposed for combining pixel and super pixel level HSI representations to extract salient features of HSI. To explore the spatial information of HSI, context aware GCN and multiscale GCN were proposed in (Wan et al., 2019) and (Wan et al., 2020), respectively. Nevertheless, since HSIs are so large that even with super-pixel segmentation, a semi-supervised GNN still leads to a massive amount of computation and limits its applicability (Hong et al., 2020).

Based on the problems of the above DL models, namely 1) high time complexity with increasing number of layers of CNNs and 2) high space complexity with using semi-supervised GNNs, in this paper, we propose the dynamic graph convolutional networks (DGCNet), as shown in Figure 1. Technically, HSI usually consists of hundreds of very narrow spectral. It contains a lot of redundant spectral information, which significantly

* Correspondence: dongyanni@cug.edu.cn; Tel.: +86–137–2019–2758

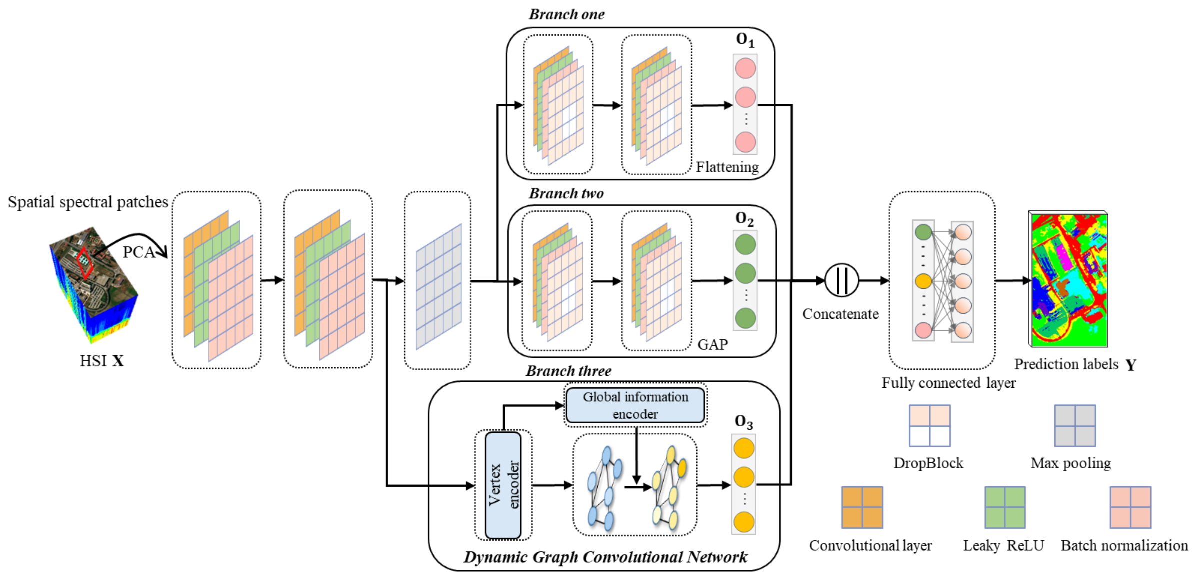

Figure 1. The architecture of the proposed DGCNet. Given an input HSI $\mathbf{X}$, our method first use two convolution blocks extraction shallow features. Next, the deep spatial-spectral features are extracted based on the Flattening and Global Average Pooling (GAP) branches. The structure feature is extracted by dynamic graph convolutional module (Sec. 3.1 & 3.2). Finally, the spatial, spectral and structure features are concatenated and classified by softmax function to obtain the label $\mathbf{Y}$ (Sec. 3.3).

increases the processing time of HSIs. Therefore, firstly, principal component analysis (PCA) is employed to extract the most informative components of the HSI. Then, two convolutional layers are employed to extract shallow features. Next, three branches extract spatial-spectral features and structural features of the HSI, respectively. Dynamic GCN, which is a novel network architecture for HSIC, is proposed to maximize the exploitation of the global structure information and further boost the performance. The module can be integrated into the head of any DL-based classification framework and trained jointly with them. Finally, we further adopt a fusing mechanism to make full use of the three features of network. In experiments on the rather challenging Indian Pines dataset, we obtain >98% overall labeling accuracy. We show that multi-different features are vital for good HSI classification and outperform standard CNN-based by a large margin.

2. RELATED WORK

2.1 Graph Convolutional Networks

Graph data are commonly found in the real world. However, the development of DL based on graph data is relatively slow due to the sequence disorder and dimensional variability of graph data (Zhou et al., 2020). Bruna et al. (Bruna et al., 2014) first proposed the concept of graph convolution based on traditional CNNs, which enabled the expansion of DL from Euclidean domain to non-Euclidean domain. Since then, graph-based DL methods have flourished along both the spatial domain and spectral domain.

Spectral approaches implement convolution operations on topological graphs with the help of graph theory. Unfortunately, the computational complexity of the graph Fourier transform, which must be used, is $O(n^2)$. Therefore, ChebNet was proposed in (Defferrard et al., 2017), which defines the Chebyshev polynomial of the diagonal matrix of the feature vector as a filter, to reduce the computational complexity. Further, first-order ChebNet (Kipf and Welling, 2017) was proposed to further reduce the computational complexity of GCNs, which is comparable to the top semi-supervised methods in terms of computational efficiency and accuracy. Subsequently, a large number of variants of GCNs have been developed based on this (Zhuang and Ma, 2018) (Xu et al., 2019).

Spatial approaches define the convolution operation directly on the connectivity of each node, essentially by continuously aggregating the neighbor information of nodes. Duvenaud et al. (Duvenaud et al., 2015) proposed a model where all nodes in a neighborhood share the same kernel weights. GraphSage (Hamilton et al., 2017) implements inductive training and testing by aggregating the information of neighboring nodes in a sampling and aggregation manner, which allows graph convolution to be easily extended to large graphs. Graph attention networks (Velicˇkovic et al., 2018) use the attention mechanism to determine the importance of each neighbor node to the central node when aggregating the neighbor information of node, avoiding complex matrix operations.

2.2 Regularization

Deep neural networks (DNNs) often face severe over-fitting problems, i.e., models have large gaps in accuracy between the training and test sets. Many regularization strategies have been devised to reduce testing errors (He et al., 2019).
Dropout (Krizhevsky et al., 2017) is to temporarily drop it from the network with a certain probability during training, it can be considered as a practical Bagging method for integrating a large number of DNNs. Ghiasi et al. (Ghiasi et al., 2018) proposed a structured Dropout: DropBlock removes consecutive units and forces the remaining units to learn more semantic information to enhance the model generalization ability. SpatialDropout (Tompson et al., 2015) deactivates the whole channel of the feature layer in order to avoid the overall change of semantic information. FractalNet (Larsson et al., 2017) proposes a regularization strategy for random deactivation of multi-branch

structures, which randomly removes branches in networks to reduce over-fitting.

Direct training of hard labels often results in over-confident models. Boosting labels can efficiently alleviate the over-fitting problem and improve the accuracy and robustness of DNNs. For example, Bootstrapping (Reed et al., 2015) improves the robustness of models by smoothing hard labels in two ways, Bootsoft and Boothard. Inception v3 (Szegedy et al., 2015) proposes a label smoothing strategy combined with uniformly distributed updated label vectors to avoid over-confidence in the correct labels. Xie et al. (Xie et al., 2016) present DisturbLabel, which randomly replaces a part of labels as incorrect values in a mini-batch. Li et al. (Li et al., 2020) use two networks to embed images and labels into a latent space separately and learn the relationship between them by deep distance measure for correcting the network.

3. METHOD

Concerning hyperspectral data, we denote it as $\mathbf{X} \in \mathbb{R}^{H \times W \times B}$, where ground truth represents $\mathbf{Y} = \{y_1, y_2, \cdots, y_c\}_{c=1}^{C} \in \mathbb{R}^{H \times W}$, where H, W, B and C denote the length, width, height and total number of categories of the HSI, respectively. The HSIC determines the category to which each pixel belongs by a classification model given the dataset. In this work, we propose the DGCNet, which mainly consists of graph convolutional layers and convolutional layers, divided into three branches. In the following sections, we will describe the framework in detail.

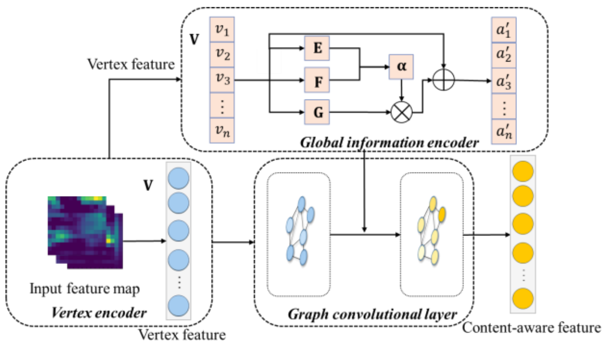

Figure 2. The flowchart of the dynamic GCN module. Given an input feature map, we first flatten the feature map to obtain vertex feature v_n. After an attention mechanism encoder, the vertex features can adaptively transform the connection relation to obtain $a_n^{'}$. The feature vector. **E,F** and **G** are obtained by fully connected network transformed. **α** is the product of **E** and. **F** Vector summation ⊕ and Vector multiplication ⊗ operations are involved in the attention mechanism encoder.

3.1 Dynamic GCN

3.1.1 Vertex Encoder

As mentioned before, GCN is able to process the information of aggregated neighbor nodes to achieve feature extraction. However, due to the incompatibility of data representation between different network architectures, it is not straightforward to integrate a GCN and a CNN directly. The feature map $\mathbf{h} \in \mathbb{R}^{H \times W \times D}$ processed by CNN first goes through vertex encoder module to obtain a set of vertex feature, each of which describes the contents related to a specific label from the input feature map. As shown in Figure 2, vertex encoder first calculates category-specific activation maps $\mathbf{M} = \{\mathbf{m}_1, \mathbf{m}_2, \cdots, \mathbf{m}_c\}_{c=1}^{C} \in \mathbb{R}^{(H \times W) \times C}$ and then they are used to convert the transformed input feature map $\mathbf{h}$ into the sequence representations $\mathbf{V} = \{\mathbf{v}_1, \mathbf{v}_2, \cdots \mathbf{v}_c\}_{c=1}^{C} \in \mathbb{R}^{C \times D}$. This can be expressed as:

$$\mathbf{v}_c = \mathbf{m}_c^T \mathbf{h}^{'} = \sum_{i=1}^{H} \sum_{j=1}^{W} m_{i,j}^{c} \mathbf{h}_{i,j}, \tag{1}$$

where $\mathbf{m}_c^T$ and $\mathbf{h}^{'} \in \mathbb{R}^{(H \times W) \times \mathrm{D}}$ are the weight of (c)th activation map and the feature vector of the feature map at (i, j), respectively. Specifically, the vertex information thus generated can selectively aggregate class-specific relevant features and use them for subsequent processing.

3.1.2 Graph Convolutional Layer

With vertex representations $\mathbf{V}$ obtained in previous section, we develop a novel dynamic GCN to adaptively extract structural information of HSI. Unlike the existing semi-supervised GCN HSIC, dynamic GCN can use supervised learning methods, and generate discriminative vectors for the final classification. Specifically, our dynamic GCN consist of two graph convolutional layers, as shown in Figure 2.

The first layer performs regular graph convolution operations. For vertex feature $\mathbf{V} \in \mathbb{R}^{C \times D}$, GCN aims to utilize a adjacency matrix $\mathbf{A}_f \in \mathbb{R}^{C \times C}$ and learnable parameters $\mathbf{W}_f$ upstate the values of $\mathbf{V}$. It is worth noting that $\mathbf{A}_f$ and $\mathbf{W}_f$ in the first layer are randomly initialized and learned by gradient in training. For the second layer, we introduce the adjacency matrix $\mathbf{A}_s$ to update the node $\mathbf{V}^{'}$. $\mathbf{A}_s$ is obtained from the encoded $\mathbf{V}$, which is different from the first layer whose is fixed after training. The $\mathbf{A}_s$ can be dynamically updated by the self-attentive mechanism as the input features change. Thus, different $\mathbf{A}_s$ are generated for each patch, which greatly enhances the expressiveness of the model and reduces the risk of over-fitting. Formally, this process can be formulated as:

$$\mathbf{V}^{l+1} = \sigma(\mathbf{A}\mathbf{V}^{l}\mathbf{W}), \tag{2}$$

where $\mathbf{A}$ denotes the adjacency matrix, and $\mathbf{W}_s$ represent the learnable parameters. $\sigma(\cdot)$ denotes the Leaky ReLU activation function. Overall, the DGCNet could capture content-aware category dependencies with the help of the dynamic graph convolutional layer.

3.2 Regularization

3.2.1 DropBlock Regularization

DropBlock is a simple regularization method like dropout. We assume that Figure (a) is a feature map. As shown in Figure 3 (b), white color indicates dropped neurons, while light blue color indicates normal neurons. Dropout randomly drops discontinuous neurons. Nevertheless, since neighboring neurons usually have similar features, the network also learns the same information from the vicinity of the dropped activation unit. Figure 3 (c) represents the random continuous regions used by DropBlock. By dropping out a portion of the whole adjacent area, the network will focus on learning other features to achieve correct classification and thus exhibit better generalization. DropBlock has two parameters which are drop block size s and drop probability γ. These two parameters together control the percentage of semantic information that is discarded.

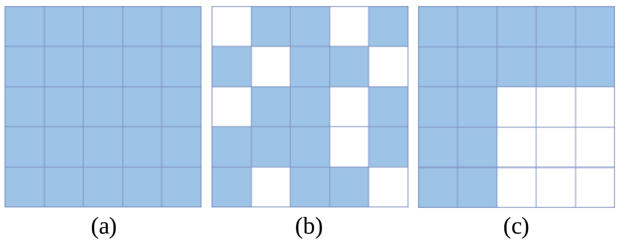
Figure 3. Schematic diagram of the DropBlock module randomly discarding semantic information. A feature map (a) is given. If light blue squares are used to indicate the feature values contained in the feature map and white squares are used to indicate the deactivated feature values, (b) represents the feature map obtained using random deactivation and (c) represents the feature map obtained using semantic deactivation.

3.2.2 Label Smoothing Regularization

Since HSIC usually employs a cross-entropy loss function. This, however, can cause two problems. First, over-fitting: The neural network will drive itself to learn in the direction with the largest difference between the correct and incorrect labels. Second, learning incorrect labels: Manual labeling will inevitably produce errors. Calculating the cross-entropy loss on the wrong labels can lead to reverse optimization results.

Label smoothing regularization adds noise by soft one-hot, which reduces the weight of the class of real sample labels in computing the loss function and ultimately has the effect of suppressing over-fitting. Considering a prior distribution of labels $u(t)$, independent of the training instance $x \in \mathbf{X}$, and a label smoothing coefficient ε. For a training instance with real labels $y \in \mathbf{Y}$, the probability distribution of the label distribution $q(t \mid x)$ after adding label smoothing becomes:

$$q^{'}(t \mid x) = (1-\varepsilon)\cdot q(t \mid x) + \varepsilon u(t)\,, \tag{3}$$

$$u(t) = \frac{1}{C}\,, \tag{4}$$

where C denotes the total number of categories, which is a mixture of the original ground truth distribution y_i and the fixed distribution $u(c)$, with weights $1-\varepsilon$ and ε, respectively.

3.3 Feature Fusion and Classification

To fully explore the information contained in HSI, we use three branches to extract HSI features from different perspectives. As shown in Figure 1, the outputs of the three branches are $\mathbf{O}_1$, $\mathbf{O}_2$ and $\mathbf{O}_3$. We flatten the feature map of the output of branch one to obtain $\mathbf{O}_1$. The output of branch two performs the global average pooling operation to get $\mathbf{O}_2$. $\mathbf{O}_3$ is obtained from the dynamic graph convolution module. The features obtained from these three different perspectives are fused and then classified. The output feature $\mathbf{O}$ can be represented as:

$$\mathbf{O} = \mathbf{O}_1 \,\|\, \mathbf{O}_2 \,\|\, \mathbf{O}_3\,, \tag{5}$$

where the operator $\oplus$ denotes concatenating features along the spectral dimension. To optimize the proposed model, we use the cross-entropy loss function as:

$$\mathcal{L} = -\frac{1}{Z}\sum_{z=1}^{Z}\sum_{c=1}^{C}\mathbf{y}_c^m \log(\tilde{\mathbf{y}}_c^m) \tag{6}$$

where Z is the number of samples in a mini-batch, and C is the total number of categories in the dataset. $\mathbf{y}_c^m$ and $\tilde{\mathbf{y}}_c^m$ are the ground truth and predict labels of the (z)th of class c, respectively. The model weights and bias are updated by back-propagation and stochastic gradient descent.

4. EXPERIMENTS

In this section, six supervised classification methods are used to compare with the proposed DGCNet, including a machine learning based benchmark algorithm RBF-SVM[2] and six CNNs methods. The approaches included in the comparison are summarized as follows.

1. SVM-RBF: SVM with an RBF kernel is implemented using the scikit-learn package.
2. 3-D CNN: 3D-CNN[4] (Hamida et al., 2018) is a classical benchmark method and includes an input layer, convolutional layer, max-pooling layer, fully connected layer, and output layer.
3. DCNN: DCNN[4] (Chen et al., 2016) adds batch normalization layers to the 3D-CNN and extends the network depth.
4. SSRN: SPRN[3] is a CNN-based spectral partitioning residual network.
5. DFFN: We follow the architecture of the 2-D CNN as used in (Song et al., 2018). DFFN[3] is a 2D-CNN network that fuses the outputs of different hierarchical layers with the help of residual structure.
6. HybridSN: HybridSN[5] (Roy et al., 2020) improves the attention of the model to salient information by adding a spatial attention module and a channel attention module.
7. SPRN: SPRN[3] (Zhang et al., 2021) splits the input spectral bands into several nonoverlapping continuous sub-bands and uses cascaded parallel improved residual blocks to extract spectral–spatial features from these sub-bands.

To make a fair comparison, the spatial patch size and dimension are set to the same for all DL-based methods, while SVM adopts serialized original data. In this work, a classical benchmark dataset, namely the Indian Pines dataset, is used to verify the effectiveness of the proposed algorithm. When the number of category samples is greater than 100, 100 samples are selected as the training set, otherwise 15 samples are selected as the training set and the rest of the sample points are used for testing. The data set is divided as shown in Table 1, which can incorporate information about the neighbors of the target pixels and is beneficial for improving the accuracy of the CNN due to spatial autocorrelation.

[2] https://github.com/zhangjinyangnwpu/HSI_Classification
[3] https://github.com/shangsw/HPDM-SPRN
[4] https://github.com/nshaud/DeepHyperX
[5] https://github.com/gokriznastic/HybridSN

No.	Class	Training	Test	Total
1	Alfalfa	15	31	46
2	Corn-notill	100	1328	1428
3	Corn-min	100	730	830
4	Corn	100	137	237
5	Grass/Pasture	100	383	483
6	Grass/Trees	100	630	730
7	Grass/pasture-mowed	15	13	28
8	Hay-windrowed	100	378	478
9	Oats	15	5	20
10	Soybeans-notill	100	872	972
11	Soybeans-min	100	2355	2455
12	Soybeans-clean	100	493	593
13	Wheat	100	105	205
14	Woods	100	1165	1265
15	Bldg-Grass-Tree-Drives	100	286	386
16	Stone-steel towers	15	78	93
	Total	1260	8989	10249

Table 1. The division of the dataset used by Indian Pines dataset.

Network Components	Parameters
Cov 1	$3\times3@32$
Cov 2	$3\times3@64$
Max pooling Layer	2×2
Branch 1	$\begin{bmatrix}3\times3@64\\3\times3@64\end{bmatrix}$
	Flatten
Branch 2	$\begin{bmatrix}3\times3@64\\3\times3@128\end{bmatrix}$
	Global average pooling
Branch 3	$\begin{bmatrix}64\\\text{class number}\end{bmatrix}$
FC 1	1024
FC 2	256
Classification Layer	Softmax

Table 2. DGCNet model parameters, where Cov denotes convolutional layer, FC denotes fully connected layer.

4.1 Experimental Settings

The network structure of DGCNet is shown in Table 2. All the activation functions in DGCNet are the leaky ReLU and use the Adam optimizer with label smoothing. The number of training iterations is 100 and batch size is 128. The initial learning rate is 0.001, which decreases to one-tenth at one-half and five-sixths of the total number of iterations, respectively. For DropBlock, we set drop block size s to 3 and drop probability γ to 0.01 because the patch size is relatively small compared with the conventional RGB image size. For label smoothing regularization, since Indian Pines dataset has 16 classes, we use $u(t)=1/16$ and $\varepsilon=0.01$. The spatial patch size is set to 19×19 and dimension is reduced to 32. All parameter settings in this paper were obtained by referring to existing work and by trial-and-error methods. Some of the parameter analyses are given below. Our experiments are implemented with Python-3.8.5 and PyTorch-1.8.1. The environment consists of an i7-10700K CPU with 32 GB and a NVIDIA GTX-2060 graphical processing unit (GPU) with 6 GB.

To alleviate the influence of random factors, all the experiments were repeated five times, and the mean values of overall accuracy (OA), the average accuracy (AA) and the Kappa coefficient are taken as the evaluation indices. Precisely, OA is the ratio of the sample size of correctly predicted categories to the total tested sample size, while AA is the average of the accuracy of all tested categories. Furthermore, The Kappa coefficient is a statistic widely used to measure the consistency of multi-classification tasks.

4.2 Parameter Analysis

Deep learning often contains a very large number of hyperparameters that can sometimes significantly affect the performance of a model. A large amount of research work has been done on the setting of hyperparameters, but usually the selection of hyperparameters is done by trial and error. In this section, the hyperparameters patch size and data dimension are analysed to give an intuitive demonstration of the selection of hyperparameters for DGCNet.

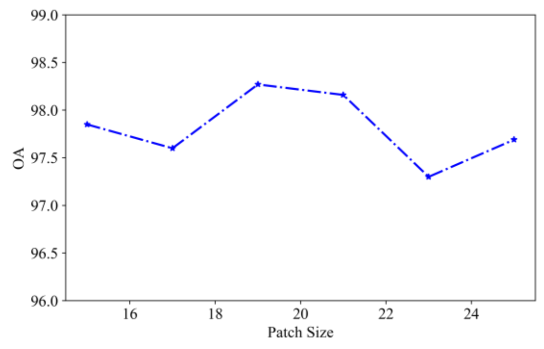

Figure. 4. Overall accuracy of DGCNet with change of patch size.

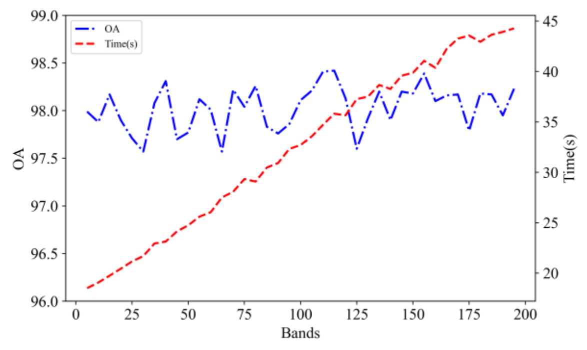

Figure 5. Overall accuracy and training time of DGCNet with change of the dimensionality.

The parameter patch size controls how much data is input to the network. As the patch size increases, more neighboring information of the target pixel is passed into the DGCNet. In HSIs, neighboring pixels usually exhibit similar spectral values. And DGCNet can extract this similar information well and use the information of neighboring pixels to assist in classification. As shown in Figure 4, we set the patch size to 15, 17, 19, 21, 23 and 25 to investigate the effect of different patch sizes on the classification results. It is interesting that the effect of patch size is not very significant. The accuracy of DGCNet on the Indian Pines dataset fluctuates between 97%-98.5%. This is probably since the dynamic graph convolution module is adaptive to model different sizes of input data. The best performance is achieved when the patch size is taken to 19, so the patch size is taken to be 19.

HSIs contain hundreds of dimensions. Each pixel can form a spectral curve. However, it is obvious that it contains very redundant information. This both increases the cost of storing hyperspectral images and significantly increases the processing time of hyperspectral images. In this paper, a simple hyperspectral dimensionality reduction method PCA is used to extract hyperspectral image principal components, which saves hyperspectral image processing time without degrading hyperspectral image classification accuracy. Figure 5 shows the schematic diagram of the variation of classification accuracy and training time as the dimensionality of hyperspectral images increases. As the hyperspectral dimension increases, the training time increases linearly from 20 s to 45 s. However, the training accuracy keeps fluctuating between 97.5% and 98.5%. This may be caused by the powerful characterization ability of deep learning, which is insensitive to the input data. Therefore, as a compromise between training accuracy and training time, we choose to downscale the hyper-spectrum to 32 dimensions.

4.3 Comparison

Table 3 shows the results of the quantitative evaluation of the Indian pine dataset. including per-class accuracy, OA, AA, kappa, training time and inference time. The proposed DGCNet outperforms all the other approaches in most cases. Specifically, SVM gets much less precision than other approaches, Showing that the importance of spatial information. The accuracy of the 3D-CNN is about 2%-4% gap compared with other space-spectral combined methods, which can be attributed to the positive impact of the architecture and specific module design. HybridSN achieved high AA, which may be due to the subtle information extraction and attention mechanism. Furthermore, since the intrinsic structure and spatial-spectral information of HSI are fully utilized, the classification accuracies obtained by DGCNet are higher than methods, and the three branches make it have the best robustness.

The training and inference time of CNNs trained by supervised learning is positively correlated with the number of samples. We also choose 100 samples to test the time efficiency of the network. As shown in Tables 1, the shallow model SVM is fast in training, but its accuracy is limited. Some of the deep models outperform SVM in inference speed due to GPU acceleration. The DGCNet uses a three-branch parallel training model, and its training and inference speed is substantially ahead of the compared deep models. For example, for the recently proposed SPRN, we only need about 1/5 of its training time to achieve similar accuracy on the Indian Pines dataset, which shows the promise of the DGCNet for applications in the industrial world.

No.	SVM	3D-CNN	DCNN	SSRN	DFFN	HybridSN	SPRN	DGCNet
1	76.13	93.02	93.69	80.13	90.56	**100.00**	75.59	85.40
2	69.37	88.46	95.75	96.81	97.66	95.37	**97.87**	97.85
3	72.38	92.74	98.03	99.04	98.45	97.37	99.17	**99.59**
4	87.59	92.24	95.76	96.89	96.06	98.19	96.63	**99.42**
5	92.01	95.41	97.31	98.12	98.04	96.91	97.43	**98.86**
6	94.67	96.67	**96.28**	93.31	93.79	93.92	93.75	94.71
7	92.31	98.81	93.55	98.57	98.57	98.57	**100.00**	**100.00**
8	97.51	98.09	99.69	98.42	98.19	99.22	99.84	99.38
9	96.00	**100.00**	93.75	79.09	79.55	75.88	66.59	92.50
10	79.79	92.01	94.13	95.45	94.51	94.97	95.90	**97.65**
11	65.80	97.46	98.88	**99.65**	99.23	98.47	99.17	99.58
12	82.39	85.48	93.57	**98.84**	98.59	95.58	97.21	97.00
13	**98.29**	96.89	98.49	92.63	92.07	95.06	91.41	96.79
14	90.35	98.72	98.93	**99.24**	97.82	98.93	99.19	99.11
15	72.87	92.22	93.11	95.29	96.62	93.38	95.54	**96.91**
16	95.38	95.46	90.20	96.85	95.26	97.92	96.74	94.11
OA(%)	78.07	94.08	96.96	97.59	97.36	96.83	97.61	**98.27**
AA(%)	85.18	94.61	95.69	94.90	95.31	95.61	93.88	**96.81**
kappa×100	75.00	93.18	96.49	97.22	96.95	96.33	97.24	**98.00**
Training time(s)	0.11	51.16	38.16	148.45	57.92	75.43	93.54	20.57
Test time(s)	1.90	1.43	0.97	1.95	1.31	1.60	1.93	0.55

Table 3. Classification maps results obtained from our proposed DGCNet and seven comparison methods on the Indian Pines dataset., including per-class accuracy, OA, AA, kappa, training time and inference time.

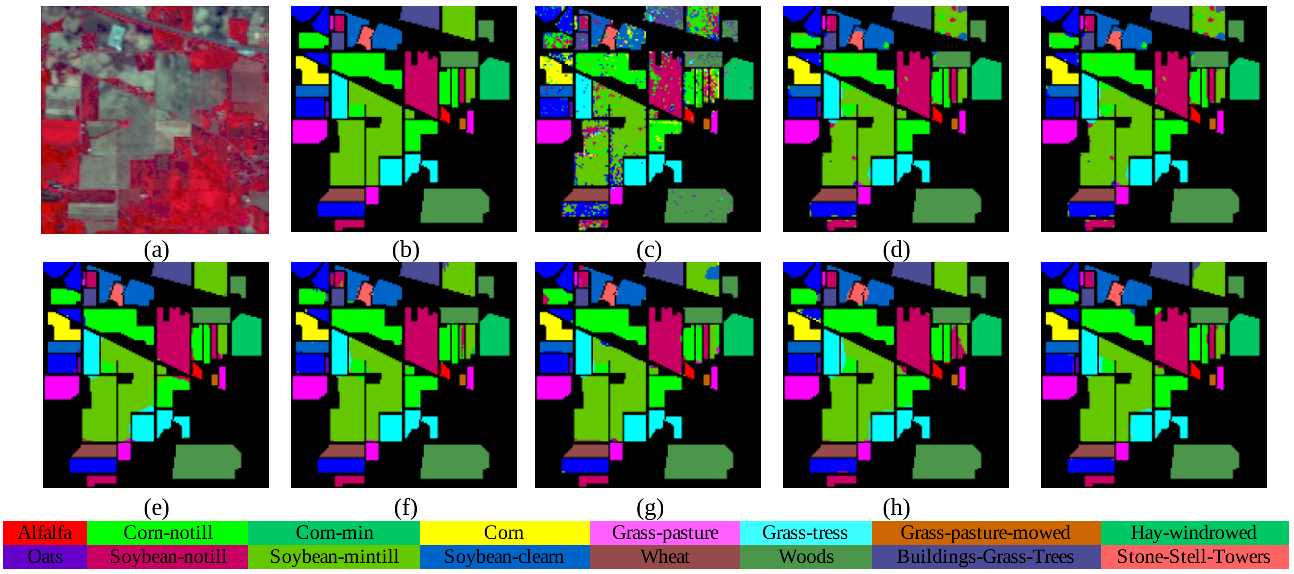

Figure 6. Classification maps results obtained from our proposed DGCNet and seven comparison methods. on the Indian Pines dataset. (a) False-color image. (b) Ground Truth. (c) SVM. (d) 3D-CNN. (e) DCNN. (f) SSRN. (g) DFFN. (h) HybridSN. (i) SPRN. (j) DGCNet.

4.4 Visualisation

More visually, Figure 6 shows the visualized classification results of all these approaches in a single experiment on the Indian Pines dataset. It can be seen from Figure 6 that the classification maps generated by DL-based methods are smoother than SVM. The classification map of SVM has serious salt and pepper noise. While our DGCNet result is both smooth and more consistent with the ground truth map than all compared methods. All the observations validate the superiority of our methods and the rationality of integrating GCN and CNN modules together.

5. CONCLUSIONS

The summary of this study summarized as given below:

This research study gives a solution to HSIC by integrating a dynamic GCN with a CNN. The framework uses three branches to process HSI in parallel which can learn from the global information, achieving faster training and inference. To alleviate the over-fitting problem caused by the small sample problem in HSIC, scheduled DropBlock is applied to learn more generalizable features and label smoothing to reduce the interference of mixed image elements on the classification performance. By combining the advantages of the dynamic GCN and CNN, the proposed DGCNet can learn features on spatial-spectral and structure information simultaneously. Experiments on the Indian Pines benchmark dataset show that the proposed framework can obtain competitive results at a faster speed compared with six other state-of-the art methods.

In addition, there are several directions to be explored in the future work. One idea is more experiments, both on further datasets and on the ablation of the model itself, to explore the properties of the DGCNet. One idea is to use a disjoint data partitioning approach, which is a test of the model generalization capability. Another possible direction is to formally state the contribution of different feature transformations to the classification results.

REFERENCES

Ahmad, M., Shabbir, S., Roy, S.K., Hong, D., Wu, X., Yao, J., Khan, A.M., Mazzara, M., Distefano, S., Chanussot, J., 2021. Hyperspectral Image Classification -- Traditional to Deep Models: A Survey for Future Prospects. arXiv:2101.06116 [cs, eess].

Bruna, J., Zaremba, W., Szlam, A., LeCun, Y., 2014. Spectral Networks and Locally Connected Networks on Graphs. arXiv:1312.6203 [cs].

Chang, C.-I., 2021. Hyperspectral Target Detection: Hypothesis Testing, Signal-to-Noise Ratio, and Spectral Angle Theories. IEEE Trans. Geosci. Remote Sens. 1–23.

Chen, Y., Jiang, H., Li, C., Jia, X., Ghamisi, P., 2016. Deep Feature Extraction and Classification of Hyperspectral Images Based on Convolutional Neural Networks. IEEE Trans. Geosci. Remote Sens. 54, 6232–6251.

Chen, Y., Lin, Z., Zhao, X., Wang, G., Gu, Y., 2014. Deep Learning-Based Classification of Hyperspectral Data. IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens. 7, 2094–2107.

Defferrard, M., Bresson, X., Vandergheynst, P., 2017. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. arXiv:1606.09375 [cs, stat].

Dong, Y., Liu, Q., Du, B., Zhang, L., 2022. Weighted Feature Fusion of Convolutional Neural Network and Graph Attention Network for Hyperspectral Image Classification. IEEE Trans. Geosci. Remote Sens. 31, 14.

Duvenaud, D., Maclaurin, D., Aguilera-Iparraguirre, J., Gómez-Bombarelli, R., Hirzel, T., Aspuru-Guzik, A., Adams, R.P., 2015. Convolutional Networks on Graphs for Learning Molecular Fingerprints. arXiv:1509.09292 [cs, stat].

Ghiasi, G., Lin, T.-Y., Le, Q.V., 2018. DropBlock: A regularization method for convolutional networks. arXiv:1810.12890 [cs].

Hamida, A.B., Benoit, A., Lambert, P., Amar, C.B., 2018. Three dimensional Deep Learning approach for remote sensing image classification. IEEE Trans. Geosci. Remote Sens. 56, 4420–4434.

Hamilton, W.L., Ying, R., Leskovec, J., 2017. Inductive Representation Learning on Large Graphs 19.

He, T., Zhang, Zhi, Zhang, H., Zhang, Zhongyue, Xie, J., Li, M., 2019. Bag of Tricks for Image Classification with Convolutional Neural Networks, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Presented at the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Long Beach, CA, USA, pp. 558–567.

He, X., Chen, Y., Ghamisi, P., 2021. Dual Graph Convolutional Network for Hyperspectral Image Classification With Limited Training Samples. IEEE Trans. Geosci. Remote Sens. 1–18.

Hong, D., Gao, L., Yao, J., Zhang, B., Plaza, A., Chanussot, J., 2020. Graph Convolutional Networks for Hyperspectral Image Classification. IEEE Trans. Geosci. Remote Sens. 1–13.

Jimenez, L.O., Landgrebe, D.A., 1998. Supervised Classification in High-Dimensional Space: Geometrical, Statistical, and Asymptotical Properties of Multivariate Data. IEEE Trans. Syst., Man, Cybern. C 28, 39–54.

Kipf, T.N., Welling, M., 2017. SEMI-SUPERVISED CLASSIFICATION WITH GRAPH CONVOLUTIONAL NETWORKS 14.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2017. ImageNet classification with deep convolutional neural networks. Commun. ACM 60, 84–90.

Larsson, G., Maire, M., Shakhnarovich, G., 2017. FractalNet: Ultra-Deep Neural Networks without Residuals. arXiv:1605.07648 [cs].

Li, C., Liu, C., Duan, L., Gao, P., Zheng, K., 2020. Reconstruction Regularized Deep Metric Learning for Multi-label Image Classification. arXiv:2007.13547 [cs, stat].

Li, Y., Zhang, H., Shen, Q., 2017. Spectral–Spatial Classification of Hyperspectral Imagery with 3D Convolutional Neural Network. Remote Sens. 9, 67.

Liang, H., Li, Q., 2016. Hyperspectral Imagery Classification Using Sparse Representations of Convolutional Neural Network Features. Remote Sens. 8, 99.

Liu, S., Marinelli, D., Bruzzone, L., Bovolo, F., 2019. A Review of Change Detection in Multitemporal Hyperspectral Images: Current Techniques, Applications, and Challenges. IEEE Geosci. Remote Sens. Mag. 7, 140–158.

Mahesh, S., Jayas, D.S., Paliwal, J., White, N.D.G., 2015. Hyperspectral Imaging to Classify and Monitor Quality of Agricultural Materials. J. Stored Products Research 61, 17–26.

Rasti, B., Hong, D., Hang, R., Ghamisi, P., Kang, X., Chanussot, J., Benediktsson, J.A., 2020. Feature Extraction for Hyperspectral Imagery: The Evolution from Shallow to Deep (Overview and Toolbox). IEEE Geosci. Remote Sens. Mag. 8, 60–88.

Reed, S., Lee, H., Anguelov, D., Szegedy, C., Erhan, D., Rabinovich, A., 2015. Training Deep Neural Networks on Noisy Labels with Bootstrapping. arXiv:1412.6596 [cs].

Roy, S.K., Krishna, G., Dubey, S.R., Chaudhuri, B.B., 2020. HybridSN: Exploring 3-D–2-D CNN Feature Hierarchy for Hyperspectral Image Classification. IEEE Geosci. Remote Sens. Lett. 17, 277–281.

Sabbah, S., 2012. Remote Sensing of Gases by Hyperspectral Imaging: System Performance and Measurements. Opt. Eng 51, 111717.

Song, W., Li, S., Fang, L., Lu, T., 2018. Hyperspectral Image Classification With Deep Feature Fusion Network. IEEE Trans. Geosci. Remote Sens. 56, 3173–3184.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z., 2015. Rethinking the Inception Architecture for Computer Vision. arXiv:1512.00567 [cs].

Tompson, J., Goroshin, R., Jain, A., LeCun, Y., Bregler, C., 2015. Efficient Object Localization Using Convolutional Networks. arXiv:1411.4280 [cs].

Velicˇkovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y., 2018. Graph Attention Networks. ICLR2018 12.

Wan, S., Gong, C., Zhong, P., Du, B., Zhang, L., Yang, J., 2020. Multiscale Dynamic Graph Convolutional Network for Hyperspectral Image Classification. IEEE Trans. Geosci. Remote Sens. 58, 3162–3177.

Wan, S., Gong, C., Zhong, P., Pan, S., Li, G., Yang, J., 2019. Hyperspectral Image Classification With Context-Aware Dynamic Graph Convolutional Network. arXiv:1909.11953 [cs, eess, stat].

Wang, W., Dou, S., Jiang, Z., Sun, L., 2018. A Fast Dense Spectral–Spatial Convolution Network Framework for Hyperspectral Images Classification. Remote Sens. 10, 1068.

Xie, L., Wang, J., Wei, Z., Wang, M., Tian, Q., 2016. DisturbLabel: Regularizing CNN on the Loss Layer, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Presented at the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Las Vegas, NV, USA, pp. 4753–4762.

Xu, B., Shen, H., Cao, Q., Qiu, Y., Cheng, X., 2019. Graph Wavelet Neural Network. arXiv:1904.07785 [cs, stat].

Zhang, S., Li, J., Li, H.-C., Deng, C., Plaza, A., 2018. Spectral–Spatial Weighted Sparse Regression for Hyperspectral Image Unmixing. IEEE Trans. Geosci. Remote Sens. 56, 3265–3276.

Zhang, X., Shang, S., Tang, X., Feng, J., Jiao, L., 2021. Spectral Partitioning Residual Network With Spatial Attention Mechanism for Hyperspectral Image Classification. IEEE Trans. Geosci. Remote Sens. 1–14.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M., 2020. Graph neural networks: A review of methods and applications. AI Open 1, 57–81.

Zhuang, C., Ma, Q., 2018. Dual Graph Convolutional Networks for Graph-Based Semi-Supervised Classification, in: Proceedings of the 2018 World Wide Web Conference on World Wide Web - WWW '18. Presented at the the 2018 World Wide Web Conference, ACM Press, Lyon, France, pp. 499–508.