HYPERSPECTRAL IMAGE CLASSIFICATION WITH HYBRIDSN MODEL AND COMPARISON OF GPU AND CPU DEPLOYMENT FOR THE TASK

PROJECT REPORT

CSE 372-INTRODUCTION TO HIGH PERFORMANCE COMPUTING

UNDER THE SUPERVISION OF

DR. RAVI SHANKAR SINGH

ASSOCIATE PROFESSOR

AND

VINOD KUMAR

RESEARCH SCHOLAR

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, IIT BHU

PROJECT SUBMITTED BY:

AKASH KUMAR GUPTA (16123003)
P. SAI PRANEETHA (16123013)
VAIBHAV KUMAR DIXIT (16123018)

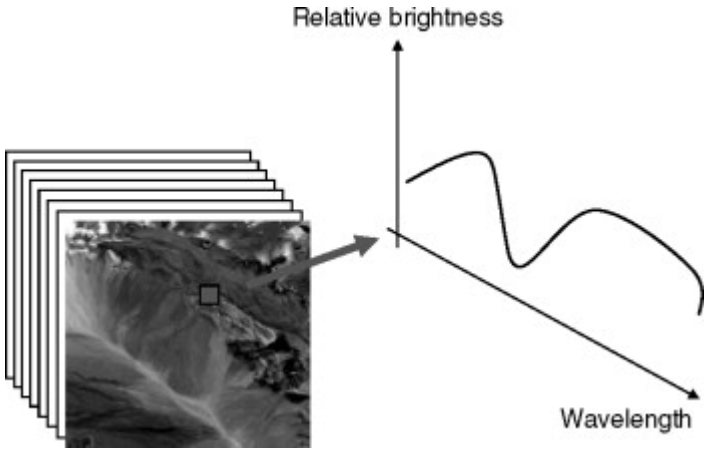
MATHEMATICS AND COMPUTING (PART-IV)

Introduction

HYPERSPECTRAL IMAGES

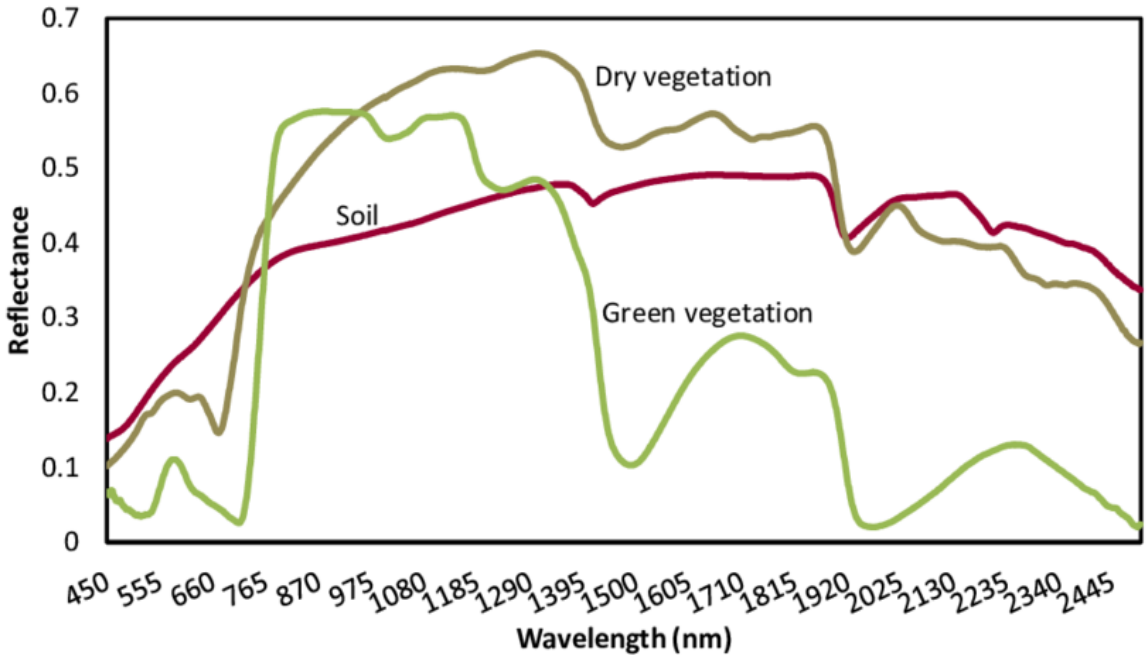
Hyperspectral images are volumetric image cubes that consist of hundreds of spatial images. Each spatial image, or spectral band, captures the responses of ground objects at a particular wavelength. Pixel values of spectral bands along the spectral direction represent the spectra of the captured objects, which can provide useful information to identify the ground objects. Scientists may use hyperspectral data to determine if a pixel may correspond to vegetation, soil, or water, which have quite distinct spectral responses. Based on the frequency locations of specific absorption bands, the wavelength ranges where the materials selectively absorb the incident energy, it may be possible to infer the molecular compositions of the objects.

Hyperspectral imagery has become an important tool for a variety of remote sensing and scanning applications; it is widely used in the sensing and discovering of ground minerals, in monitoring of the Earth's resources, and in military surveillance.



SPECTRAL SIGNATURE

Spectral signatures are the combination of reflected, absorbed and transmitted or emitted EMR by objects at varying wavelengths, which can uniquely identify an object. When the amount of EMR (usually intensity of reflected radiation or reflectance in percentage) coming from the material is plotted over a range of wavelengths, the connected points produce a curve which is known as spectral signature of the material or in other words spectral response curve. To interpret remote sensing images, it is absolutely important 42 Introduction to Remote Sensing to start with a basic understanding of spectral signature, which means how different terrain features such as water, rock, soils, and vegetation, interact with the different wavelengths (bands) of the EMR.



CONVOLUTION NEURAL NETWORK

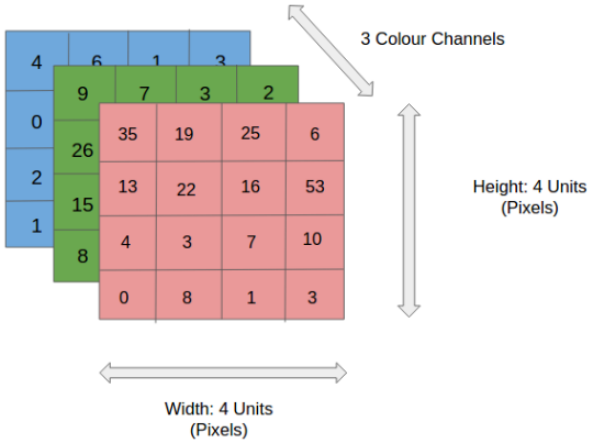
A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area.

We can imagine how computationally intensive things would get once the images reach dimensions, say 8K (7680×4320). The role of the ConvNet is to reduce the images into a form which is easier to process, without losing features which are critical for getting a good prediction. This is important when we are to design an architecture which is not only good at learning features but also is scalable to massive datasets.

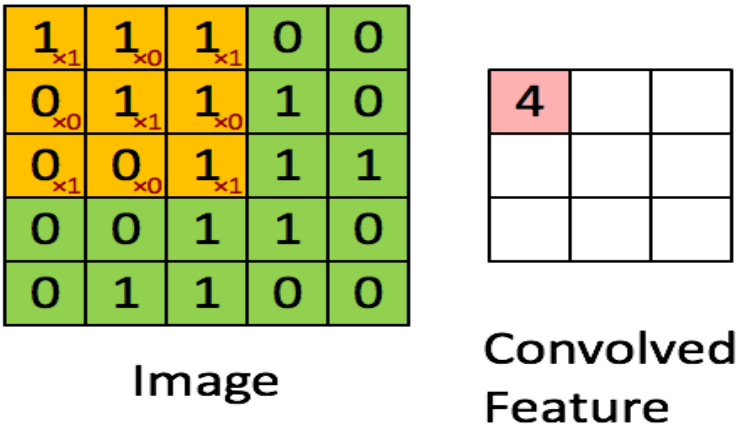
The objective of the Convolution Operation is to extract the high-level features such as edges, from the input image. ConvNets need not be limited to only one Convolutional Layer. Conventionally, the first ConvLayer is responsible for capturing the Low-Level features such as edges, color, gradient orientation, etc. With added layers, the architecture adapts to the High-Level features as well, giving a network which has the wholesome understanding of images in the dataset, similar to how we would.

INPUT IMAGE



In the figure, we have an RGB image which has been separated by its three color planes — Red, Green, and Blue. There are a number of such color spaces in which images exist — Grayscale, RGB, HSV, CMYK, etc.

CONVOLUTION LAYER-THE KERNEL



In the above demonstration, the green section resembles our 5x5x1 input image, I. The element involved in carrying out the convolution operation in the first part of a Convolutional Layer is called the Kernel/Filter, K, represented in the colour yellow. We have selected K as a 3x3x1 matrix.

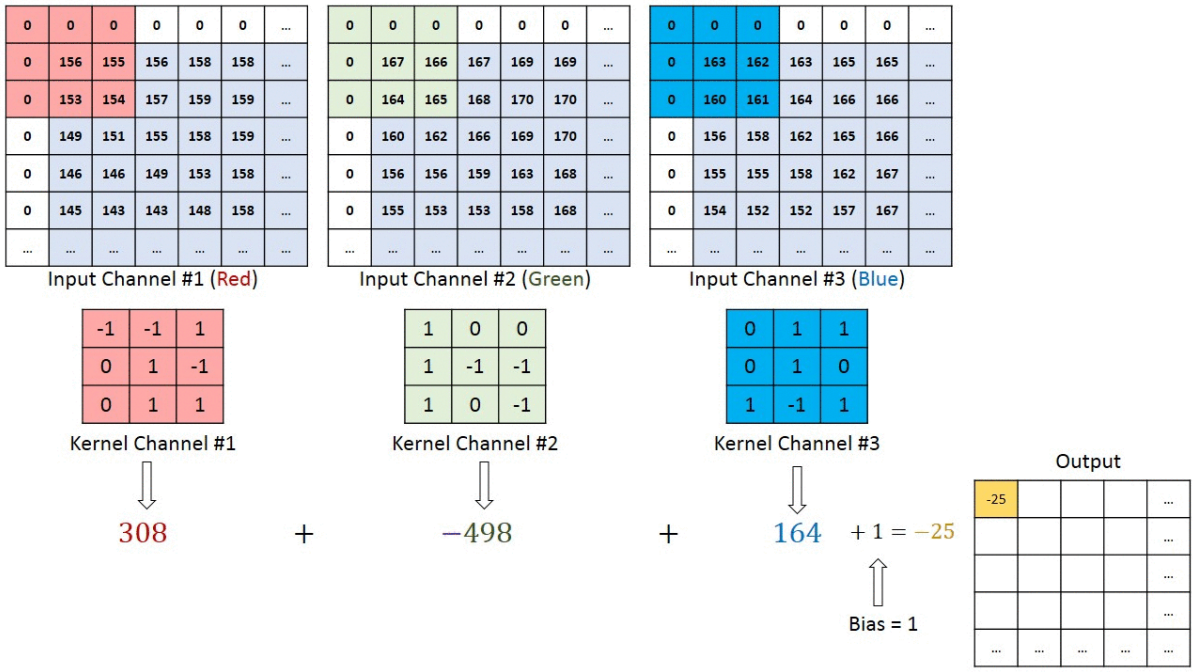
STRIDE

Stride is the number of pixels shifts over the input matrix. When the stride is 1 then we move the filters to 1 pixel at a time. When the stride is 2 then we move the filters to 2 pixels at a time and so on.

In the above example, the Kernel shifts 9 times because of Stride Length = 1 (Non-strided), every time performing a matrix multiplication operation between K and the portion P of the image over which the kernel is hovering. The filter moves to the right with a certain Stride Value till it parses the complete width. Moving on, it hops down to the beginning (left) of the image with the same Stride Value and repeats the process until the entire image is traversed.

MULTIPLE CHANNEL INPUT

In the case of images with multiple channels (e.g. RGB), the Kernel has the same depth as that of the input image. Matrix Multiplication is performed between Kn and In stack ([K1, I1]; [K2, I2]; [K3, I3]) and all the results are summed with the bias to give us a squashed one-depth channel Convoluted Feature Output.



PADDING

It's an additional layer that we can add to the border of an input image. There are two types of results to the operation — one in which the convolved feature is reduced in dimensionality as compared to the input, and the other in which the dimensionality is either increased or remains the same. This is done by applying Valid Padding in case of the former, or Same Padding in the case of the latter.

SAME PADDING- When we augment the 5x5x1 image into a 6x6x1 image and then apply the 3x3x1 kernel over it, we find that the convolved matrix turns out to be of dimensions 5x5x1. Hence the name - Same Padding.

VALID PADDING- If we perform the same operation without padding, we are presented with a matrix which has dimensions of the Kernel (3x3x1) itself -Valid Padding.

POOLING LAYER

The Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to decrease the computational power required to process the data through dimensionality reduction. Furthermore, it is useful for extracting dominant features which are rotational and positional invariant, thus maintaining the process of effectively training of the model. There are two types of Pooling: Max Pooling and Average Pooling-

MAX POOLING- Max Pooling returns the maximum value from the portion of the image covered by the Kernel. Max Pooling also performs as a Noise Suppressant. It discards the noisy activations altogether and also performs de-noising along with dimensionality reduction.

AVERAGE POOLING- Average Pooling returns the average of all the values from the portion of the image covered by the Kernel. Average Pooling simply performs dimensionality reduction as a noise suppressing mechanism.

CLASSIFICATION-FULLY CONNECTED LAYER (FC LAYER)

Adding a Fully-Connected layer is a (usually) cheap way of learning non-linear combinations of the high-level features as represented by the output of the convolutional layer. The Fully-Connected layer is learning a possibly non-linear function in that space.

We have converted our input image into a suitable form for our Multi-Level Perceptron, we shall flatten the image into a column vector. The flattened output is fed to a feed-forward neural network and backpropagation applied to every iteration of training. Over a series of epochs, the model is able to distinguish between dominating and certain low-level features in images and classify them using the Softmax Classification technique.

There are various architectures of CNNs available which have been key in building algorithms which power and shall power AI as a whole in the foreseeable future. Some of them have been listed below:

1. AlexNet
2. VGGNet
3. GoogLeNet
4. ResNet
5. ZFNet

One-Dimensional CNN(1D-CNN)

The architecture of CNN is different from other deep learning models. There are two special aspects in the architecture of CNN, i.e., local connections and shared weights. CNN exploits the local correlation using local connectivity between the neurons of near layers.

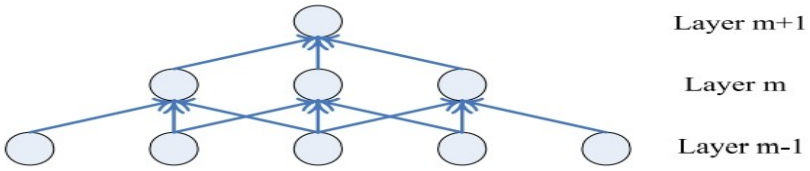


Fig. 1. Local connections in the architecture of the CNN.

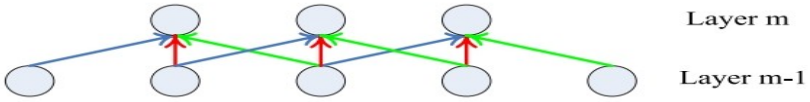


Fig. 2. Shared weights in the architecture of the CNN.

We illustrate this in Fig. 1, where the neurons in the m^{th} layer are connected to three adjacent neurons in the $(m- 1)^{th}$ layer, as an example. In CNN, some connections between neurons are replicated across the entire layer, which share the same weights and biases. In Fig. 2, the same colour indicates the same weight. Using a specific architecture like local connections and shared weights, CNN tends to provide better generalization when facing computer vision problems. A complete CNN stage contains a convolution layer and a pooling layer. Deep CNN is constructed by stacking several convolution layers and pooling layers to form deep architecture. The convolutional layer is

introduced first. The value of a neuron v^x_{ij} at position x of the j^{th} feature map in the i^{th} layer is denoted as follows:

$$v_{ij}^{x} = g\left(b_{ij} + \sum_{m}\sum_{p=0}^{P_i-1} w_{ijm}^{p} v_{(i-1)m}^{x+p}\right) \qquad (1)$$

$$g(x) = \tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \qquad (2)$$

Where m indexes the feature map in the previous layer ($(i-1)^{th}$ layer) connected to the current feature map and w^p_{ijm} is the weight of position p connected to the m^{th} feature map.

Pi is the width of the kernel toward the spectral dimension, and bij is the bias of jth feature map in the ith layer.

Pooling can offer invariance by reducing the resolution of the feature maps Each pooling layer corresponds to the previous convolutional layer. The neuron in the pooling layer combines a small N × 1 patch of the convolution layer. The most common pooling operation is max pooling, which is used throughout this paper.

The max pooling is as follows:

$$a_j = \max_{N\times 1}\left(a_i^{n\times 1} u(n,1)\right) \qquad (3)$$

where u(n, 1) is a window function to the patch of the convolution layer, and aj is the maximum in the neighbourhood.

All layers, including the convolutional layers and pooling layers of the deep CNN model, are trained using a backpropagation algorithm.

Two-Dimensional CNN(2D-CNN)

A complete 2-D CNN layer contains a convolutional layer and a pooling layer the input data is convolved with 2D kernels. The 2-D convolutional layer is obtained by the extension of (1). The convolution happens by computing the sum of the dot product between input data and kernel. The kernel is strided over the input data to cover full spatial dimension. The convolved features are passed through the activation function to introduce the non-linearity in the model. The

activation value of a neuron v^{xy}_{ij} at position (x, y) of the j^{th} feature map in the ith layer is denoted as follows:

$$v_{ij}^{xy} = g\left(b_{ij} + \sum_{m}\sum_{p=0}^{P_i-1}\sum_{q=0}^{Q_i-1} w_{ijm}^{pq} v_{(i-1)m}^{(x+p)(y+q)}\right) \quad (7)$$

where m indexes the feature map in the $(i - 1)^{th}$ layer connected to the current (j^{th}) feature map, w^{pq}_{ijm} is the weight of position (p, q) connected to the mth feature map, Pi and Qi are the height and the width of the spatial convolution kernel, and b_{ij} is the bias of the j^{th} feature map in the ith layer.

Pooling is carried out in the similar way to the 1-D CNN. The neuron in the pooling layer combines a small n × n patch of the convolutional layer.

Fine-Tuning and Classification

Based on the theory described previously, a variety of CNN architectures can be developed. In this section, we present the designed CNN for a single band (the first principal component) of HSI. The architecture is shown in Fig. 4.

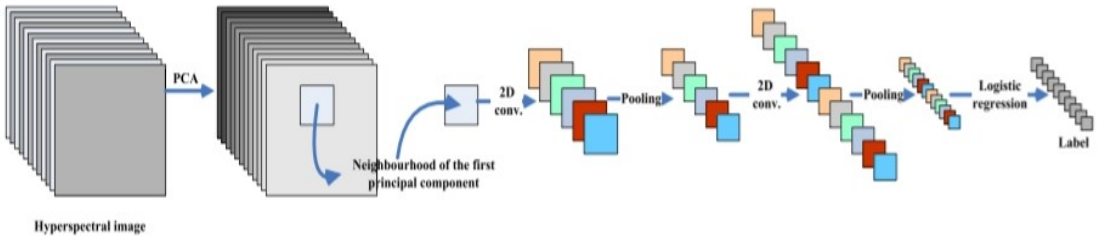


Fig. 4. Architecture of CNN with spatial features for HSI classification. The first step of processing is PCA along with spectral dimension, and then CNN is introduced to extract layerwise deep features.

We choose K × K neighbourhoods of a current pixel as the input to the 2-D CNN model. Then, we build deep CNN to extract the useful features. Each layer of CNN contains 2-D convolution and pooling. When the spatial resolution of the image is not very high, 4 × 4 kernel or 5 × 5 kernel can be selected to run convolution and 2 × 2 kernel for pooling. After several layers of convolution and pooling, the input image can be represented by some feature vectors, which capture the

spatial information contained in the K × K neighbourhood region of the input pixel. Then, the learned features are fed to the LR for classification.

Three-Dimensional CNN(3D-CNN)

The 2-D CNN extracts the local spatial features of each pixel. We further develop 3-D CNN to learn both spatial and spectral features of HSI. Fig. 5 shows the comparison of 2-D and 3-D convolutions.

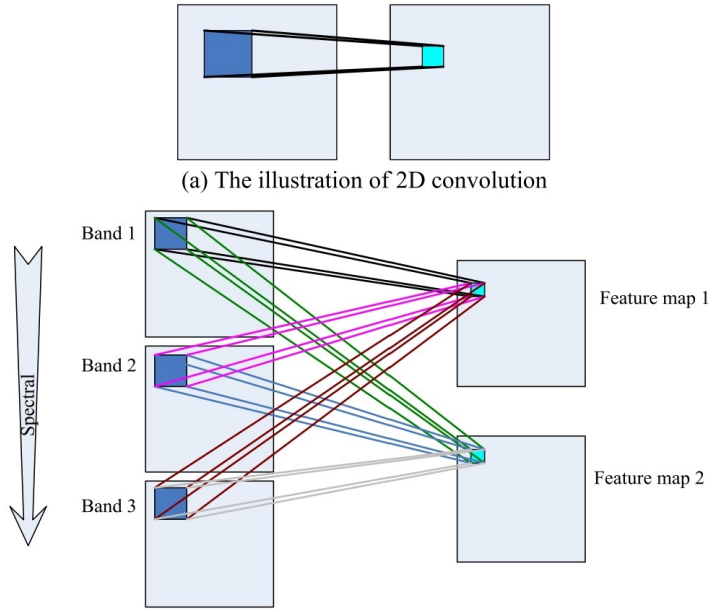


Fig. 5. Comparison of (a) 2-D and (b) 3-D convolutions. In (b), the size of the convolution kernel in the spectral dimension is 3 and the weights are color-coded.

Fig. 5. Comparison of (a) 2-D and (b) 3-D convolutions. In (b), the size of the convolution kernel in the spectral dimension is 3 and the weights are color-coded.

The 3D convolution is done by convolving a 3D kernel with the 3D-data. In the proposed model for HSI data, the feature maps of convolution layer are generated using the 3D kernel over multiple contiguous bands in the input layer; this captures the spectral information. The activation value of a neuron v^{xyz}_{ij} at position (x, y, z) of the j^{th} feature map in the ith layer is given by

$$v_{ij}^{xyz} = g\left(\sum_{m}\sum_{p=0}^{P_i-1}\sum_{q=0}^{Q_i-1}\sum_{r=0}^{R_i-1} w_{ijm}^{pqr} v_{(i-1)m}^{(x+p)(y+q)(z+r)} + b_{ij}\right) \quad (8)$$

where m indexes the feature map in the $(i - 1)^{th}$ layer connected to the current (j^{th}) feature map, and Pi and Qi are the height and the width of the spatial convolution kernel. R_i is the size of the kernel along toward spectral dimension, w^{pqr}_{ijm} is the value of position (p, q, r) connected to the m^{th} feature map, and b_{ij} is the bias of the j^{th} feature map in the ith layer.

Through 3-D convolution, CNN can extract the spatial and spectral information of hyperspectral data simultaneously. The learned features are useful for the further image classification step.

Spectral-Spatial FE Framework

Hyperspectral remote sensing images contain both spatial and spectral information. In this section, we integrate the spectral and spatial features together to construct a joint spectral–spatial classification framework using a 3-D CNN.

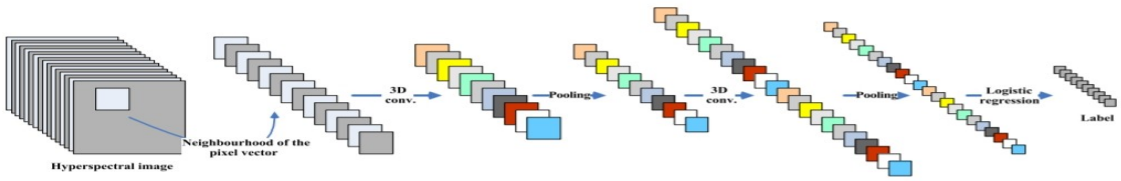


Fig. 6. Architecture of 3-D CNN with spectral–spatial features for HSI classification.

Fig. 6 shows the architecture of 3-D CNN for HSI classification. We choose K × K × B neighbourhoods of a pixel as an input to the 3-D CNN model, in which B is the number of bands. Each layer of CNN contains 3-D convolution and pooling.

As an example, a 4 × 4 × 32 kernel or a 5 × 5 ×32 kernel can be applied to 3-D convolution, and a 2 × 2 kernel can be applied for subsampling. After performing a deep 3-D CNN, the LR approach is conducted for the classification step.

Method

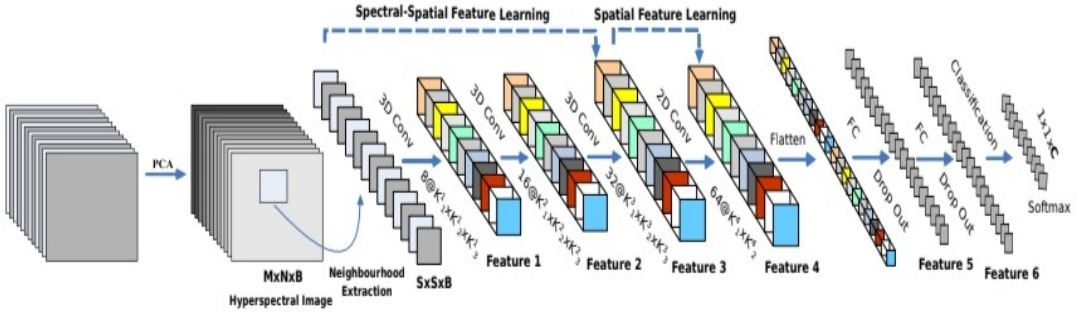
HybridSN Model

Let the spectral-spatial hyperspectral data cube be denoted by $I \in R^{M \times N \times D}$, where I is the original input, M is the width, N is the height, and D is the number of spectral bands/depth. Every HSI pixel in I contains D spectral measures and forms a one-hot label vector Y = (y1, y2, . . . yC) $\in R^{1 \times 1 \times C}$, where C represents the land-cover categories. However, the hyperspectral pixels exhibit the mixed land-cover classes, introducing the high intra-class variability and interclass similarity into I.

To remove the spectral redundancy first the traditional principal component analysis (PCA) is applied over the original HSI data (I) along spectral bands. The PCA reduces the number of spectral bands from D to B while maintaining the same spatial dimensions (i.e., width M and height N).

We have reduced only spectral bands such that it preserves the spatial information which is very important for recognising any object. We represent the PCA reduced data cube by $X \in R^{M \times N \times B}$, where X is the modified input after PCA, M is the width, N is the height, and B is the number of spectral bands after PCA.

In order to utilize the image classification techniques, the HSI data cube is divided into small overlapping 3D-patches, the truth labels of which are decided by the label of the centred pixel. We have created the 3D neighbouring patches $P \in R^{S \times S \times B}$ from X, centred at the spatial location (α, ß), covering the S×S window or spatial extent and all B spectral bands. The total number of generated 3D-patches (n) from X is given by (M − S + 1) × (N − S + 1). Thus, the 3D-patch at location (α, ß), denoted by Pα,ß, covers the width from α − (S − 1)/2 to α + (S − 1)/2, height from ß − (S − 1)/2 to ß + (S − 1)/2 and all B spectral bands of PCA reduced data cube X.



In order to take the advantages of the automatic feature learning capability of both 2D and 3D CNN, we propose a hybrid feature learning framework called

HybridSN for HSI classification. The flow diagram of the proposed HybridSN network is shown in above figure 7. It comprises of three 3D convolutions (Eqn. 2), one 2D convolution (Eqn. 1) and three fully connected layers.

In HybridSN framework, the dimensions of 3D convolution kernels are

8×3×3×7×1 (i.e., $K1_1 = 3$, $K1_2 = 3$, and $K1_3 = 7$ in Fig. 7),

16×3×3×5×8 (i.e., $K2_1 = 3$, $K2_2 = 3$, and $K2_3 = 5$ in Fig. 7) and

32×3×3×3×16 (i.e., $K3_1 = 3$, $K3_2 = 3$, and $K3_3 = 3$ in Fig. 7)

in the subsequent 1^{st} , 2^{nd} and 3^{rd} convolution layers, respectively, where 16×3×3×5×8 means 16 3D-kernels of dimension 3×3×5 (i.e., two spatial and one spectral dimension) for all 8 3D input feature maps. Whereas, the dimension of 2D convolution kernel is 64 × 3 × 3 × 576 (i.e., $K4_1 = 3$ and $K4_2 = 3$ in Fig. 7), where 64 is the number of 2D-kernels, 3 × 3 represents the spatial dimension of 2D-kernel, and 576 is the number of 2D input feature maps. To increase the number of spectral-spatial feature maps simultaneously, 3D convolutions are applied thrice and can preserve the spectral information of input HSI data in the output volume. The 2D convolution is applied once before the flatten layer by keeping in mind that it strongly discriminates the spatial information within the different spectral bands without substantial loss of spectral information, which is very important for HSI data. A detailed summary of the proposed model in terms of the layer types, output map dimensions and number of parameters is given in Table I. It can be seen that the highest number of parameters are present in the 1^{st} dense layer. The number of node in the last dense layer is 16, which is same as the number of classes in Indian Pines dataset. Thus, the total number of parameters in the proposed model depends on the number of classes in a dataset. The total number of trainable weight parameters in HybridSN is 5, 122, 176 for Indian Pines dataset. All weights are randomly initialised and trained using back-propagation algorithm with the Adam optimiser by using the softmax loss. We use mini batches of size 256 and train the network for 100 epochs with no batch normalization and data augmentation.

Results:

Dataset:

Indian Pines:

This scene was gathered by AVIRIS sensor over the Indian Pines test site in North-western Indiana and consists of 145 X145 pixels and 224 spectral reflectance bands in the wavelength range 0.4–2.5 10^(-6) meters. This scene is a subset of a larger one. The Indian Pines scene contains two-thirds agriculture, and one-third forest or other natural perennial vegetation. There are two major dual lane highways, a rail line, as well as some low density housing, other built structures, and smaller roads. Since the scene is taken in June some of the crops present, corn, soybeans, are in early stages of growth with less than 5% coverage. The ground truth available is designated into sixteen classes and is not all mutually exclusive. We have also reduced the number of bands to 200 by removing bands covering the region of water absorption: [104-108], [150-163], 220. Indian Pines data are available through Pursue's university MultiSpec site.

Model:

Model: "model"

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 25, 25, 30, 1)]	0
conv3d (Conv3D)	(None, 23, 23, 24, 8)	512
conv3d_1 (Conv3D)	(None, 21, 21, 20, 16)	5776
conv3d_2 (Conv3D)	(None, 19, 19, 18, 32)	13856
reshape (Reshape)	(None, 19, 19, 576)	0
conv2d (Conv2D)	(None, 17, 17, 64)	331840
flatten (Flatten)	(None, 18496)	0
dense (Dense)	(None, 256)	4735232

dropout (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 128)	32896
dropout_1 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 16)	2064

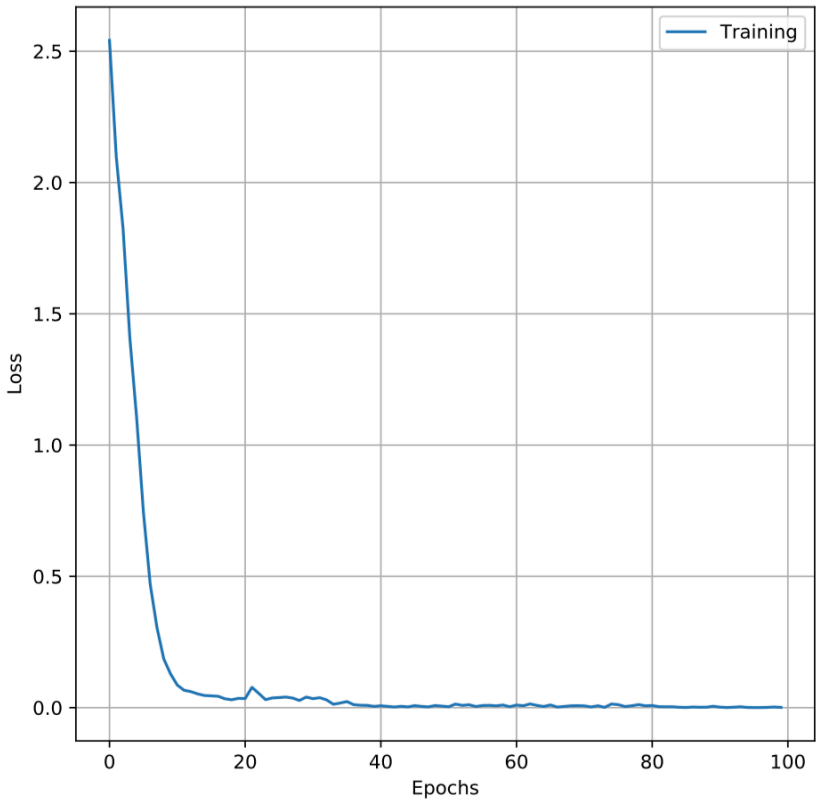
Total params: 5,122,176
Trainable params: 5,122,176
Non-trainable params: 0

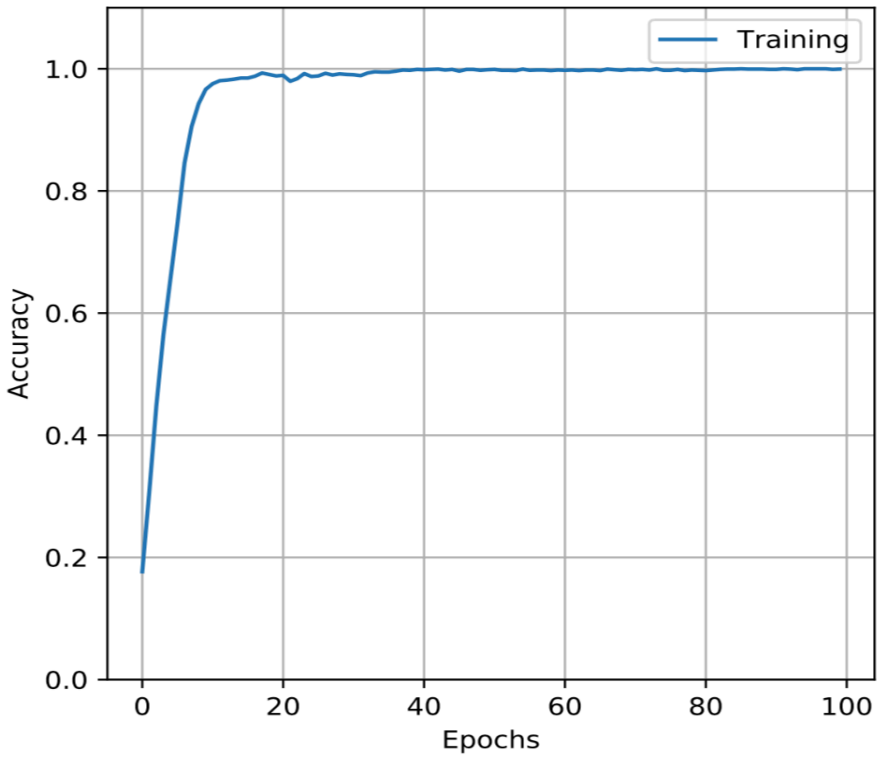
Predicted Classification

Ground Truth

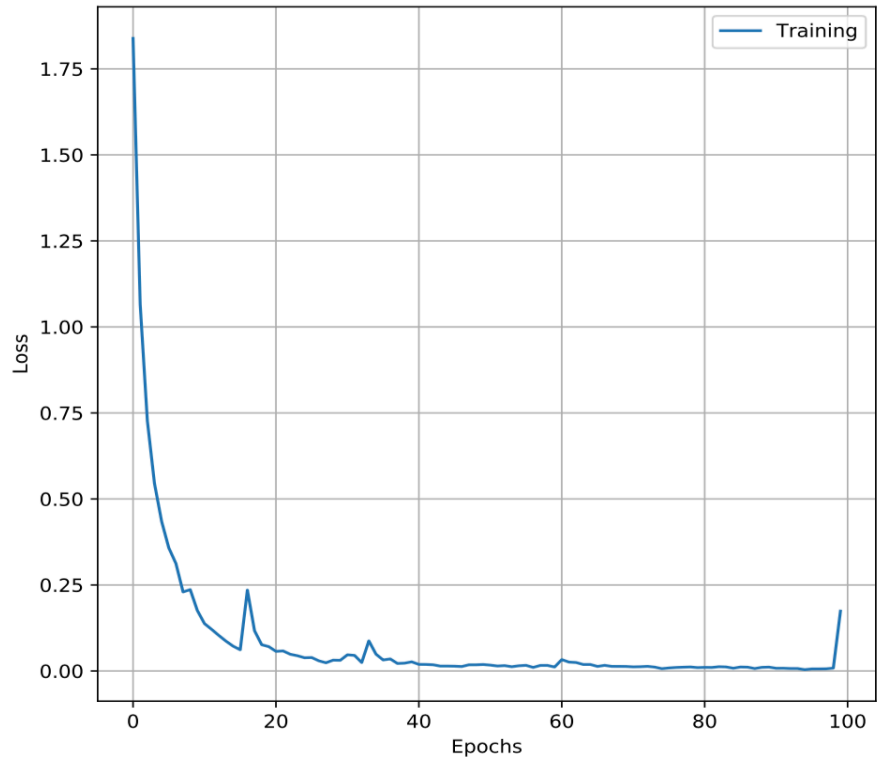
Comparison between CPU and GPU runs:

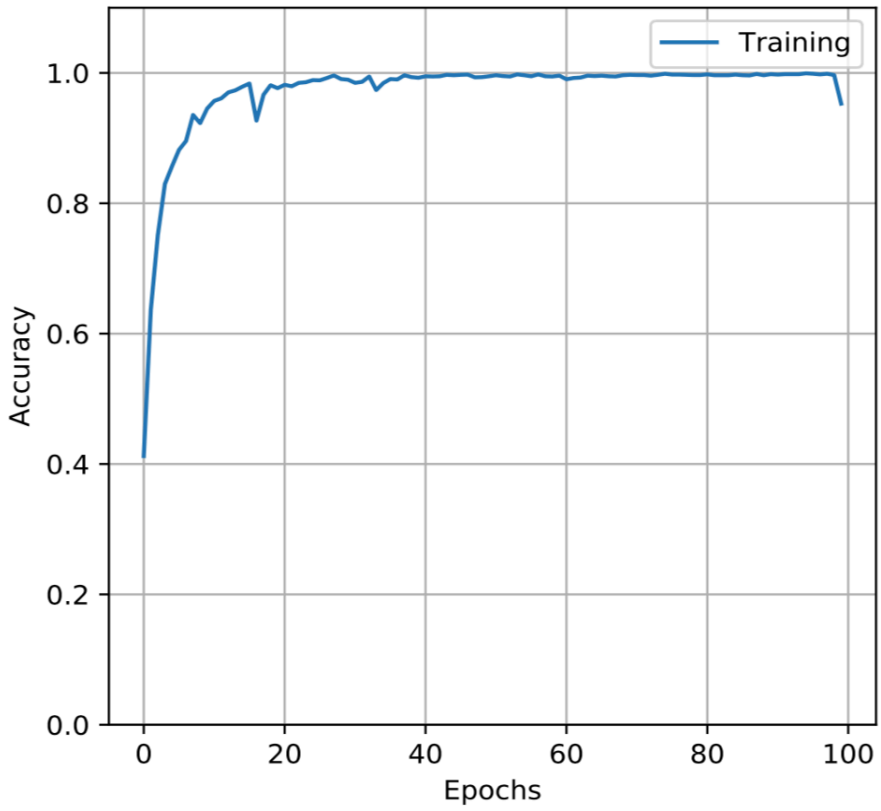
Number of GPUs used - 1
Total gpu time: 171.15567111968994





Number of CPUs used - 1
Total cpu time: 19263.765327215195





Acknowledgements

We would like to express heartfelt gratitude to Dr Ravi Shankar Singh and Mr. Vinod Kumar for their constant guidance and support.