SPECTRAL-SPATIAL CLASSIFICATION OF HYPERSPECTRAL REMOTE SENSING IMAGES USING VARIATIONAL AUTOENCODER AND CONVOLUTION NEURAL NETWORK

A. Belwalkar[1,*], A. Nath[1], O. Dikshit[1]

[1] Dept. of Civil Engineering, Indian Institute of Technology Kanpur, India – (anirudhb, avishekn, onkar)@iitk.ac.in

Commission V, SS: Emerging Trends in Remote Sensing

KEYWORDS: Hyperspectral, classification, feature extraction, spectral channels, deep learning

ABSTRACT:

In this paper, we propose a spectral-spatial feature extraction framework based on deep learning (DL) for hyperspectral image (HSI) classification. In this framework, the variational autoencoder (VAE) is used for extraction of spectral features from two widely used hyperspectral datasets- Kennedy Space Centre, Florida and University of Pavia, Italy. Additionally, a convolutional neural network (CNN) is utilized to obtain spatial features. The spatial and spectral feature vectors are then stacked together to form a joint feature vector. Finally, the joint feature vector is trained using multinomial logistic regression (softmax regression) for prediction of class labels. The classification performance analysis is done through generation of the confusion matrix. The confusion matrix is then used to calculate Cohen’s Kappa (*K*) to get a quantitative measure of classification performance. The results show that the *K* value is higher than 0.99 for both HSI datasets.

1. INTRODUCTION

Hyperspectral imaging sensors have narrow continuous spectral channels from visible to IR wavelengths for the same spatial location. The information obtained from these sensors is in the form of grids where each element represents a pixel vector. The size of each vector is equivalent to the number of spectral channels or bands. Accurate spectroscopic information obtained from hyperspectral sensors increases the ability to distinguish between different land-cover classes with enhanced accuracy. Various hyperspectral imaging systems are currently available, providing an enormous amount of image data that finds application in diverse fields such as forestry, ecology, geology, hydrology, precision farming and military applications. These applications often require class recognition of each pixel with a small number of pixels as training samples.

There are several important issues when it comes to HSI classification. Supervised classification methods face problems associated with an imbalance between the high dimensionality (Donoho et al., 2000) of data and limited access to the available training samples, the presence of highly correlated bands, and the presence of mixed pixels (Ghamisi et al., 2017). When dimensionality (number of bands) grows, keeping the number of training samples constant, the accuracy of statistical evaluation is reduced which leads to a reduction in the classification accuracy beyond some bands (Landgrebe, 2003). For classification, these problems relate to the so called curse of dimensionality (Hughes, 1968). Also, with the increased spatial resolution of new hyperspectral imaging sensors, the intra-class variation increases and the inter-class- variation decreases both in the spatial and spectral domain, leading to lower interpretation accuracies. Due to these issues with HSI classification, it is necessary to utilize spatial information and to reduce data dimensionality without losing desirable information.

In HSI, dimensionality reduction has two main categories, i.e., band selection and feature extraction. In-band selection, we obtain a subset of the original bands by minimizing spectral redundancy (Yang et al., 2011; Du et al., 2008). The process selects an appropriate band subset from the original bands, while feature extraction methods preserve essential features through mathematical transformations. Some of the most common feature extraction method for HSI include principal component analysis (PCA) (Rodarmel and Shan, 2002), Fisher’s Linear discriminant analysis (FLDA) (Bandos et al., 2009) and Independent component analysis (IDA) (Villa et al., 2011). All these methods project the original data into a low-dimensional subspace. However, hyperspectral images are considered inherently non-linear (Bachmann et al., 2005). These linear methods are not suitable for the analysis of such data (Han and Goodenough, 2008).

HSI classification is one of the active areas in the remote sensing community. Till the advent of deep learning (DL) in remote sensing, researchers have used pertinent machine learning algorithms prevalent in computer vision tasks for classifying HSI. Gaulteri (1998) carried out classification of HSI using SVM. Melagni and Bruzzone (2004) compared the classification performance of SVM with two non-parametric classifiers, k-NN and RBF kernel based ANN. Chen et al. (2014) first used DL in HSI classification where they extracted joint spectral-spatial features using multi-layered stacked autoencoders. For classification, a logistic regression based classifier was used which achieved 97-98% accuracy on the Pavia dataset. Hu et al. (2015) then proposed a simple CNN structure with five layers: one input, one convolutional, one max pooling, one fully connected and one output layer for spectral classification of HSI which achieved 90-92% overall accuracy for India Pines, Salinas and Pavia University datasets. Later, Chen et al. (2016) proposed a 3D CNN which could extract robust feature vectors by exploiting both spectral and spatial features of HSI. Mou et al. (2017) used RNN with parametric rectified tanh activation

[*] Corresponding Author

function for the first time in HSI data classification which tried to employ the sequential property of HSI data for classification.

The rest of this paper is organized as follows: Section 2 presents the description of VAE that is used for extraction of spectral features. Section 3 presents the description of CNN and multinomial logistic regression that are used for spatial feature extraction and fine tuning of pre-trained model, respectively. The spectral-spatial feature extraction framework for HSI classification is introduced in section 4. The experimental results are reported in Section 5. We conclude this paper in Section 6 with some discussions.

2. VAE BACKGROUND

2.1 Vanilla autoencoder

A Vanilla autoencoder (figure 1) is an unsupervised neural network that attempts to encode input samples into some latent-space representation so that the inputs can be rebuilt from these representations with minimal reconstruction error. It consists of three components- encoder, the latent feature vector, and decoder. The encoder transforms the input into a latent feature vector having lower dimensions than the original input (a form of compression) through an encoding function. The decoder then tries to reconstruct the initial input by using the latent feature vector through a decoding function. Autoencoder is a feed-forward network which consists of two layers- a hidden layer and an output layer where the input and output layer have the same number of neurons. The hidden layer has less number of neurons than the input layer which forces the network to learn good representations of the input. The objective of an autoencoder is to regenerate the inputs with minimum reconstruction error.

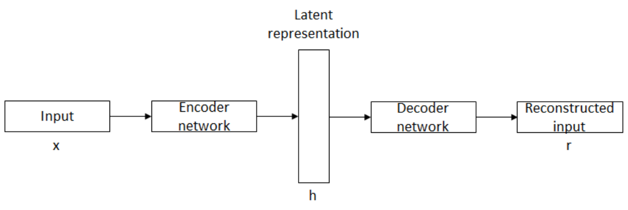

Figure 1. Vanilla autoencoder

Suppose we have a set of '*m*' data points, each having *d* dimensions i.e. $\{x^{(1)},....,x^{(i)},....,x^{(m)}\}$ and $x^{(i)} \in R^d$. Each training sample $x^{(i)}$ is encoded to a latent space representation $h^{(i)}$ using an encoding function. The latent space representation is converted back to original training sample $\hat{x}^{(i)}$ by using a decoding function. In neural network terminology,

$$h^{(i)} = f(W_1 x^{(i)} + b_1) \qquad (1)$$

$$\hat{x}^{(i)} = g(W_2 h^{(i)} + b_2) \qquad (2)$$

In the above equations W_1, W_2 denote input to hidden and hidden to output weights, respectively. *b₁.b₂* denotes the bias of hidden and output units, and *f(.),g(.)* denotes the activation function for the encoder and decoder, respectively. As the goal of an autoencoder is to have $\hat{x}^{(i)}$ approximate $x^{(i)}$, the objective function is set up as the sum of squared difference between $\hat{x}^{(i)}$ and $x^{(i)}$ which is also called reconstruction loss:

$$\left.\begin{aligned} J(W_1,b_1,W_2,b_2) &= \sum_{i=1}^{m}(\hat{x}^{(i)} - x^{(i)})^2 \\ &= \sum_{i=1}^{m}(g(W_2 h^{(i)} + b_2) - x^{(i)})^2 \\ &= \sum_{i=1}^{m}(g(W_2 f(W_1 x^{(i)} + b_1) + b_2) - x^{(i)})^2 \end{aligned}\right\} \qquad (3)$$

which can be minimized using the stochastic gradient descent algorithm. (Amari, 1993)

2.2 Variational autoencoder

The Variational autoencoder (Kingma and Welling, 2013) is a generative model (Ng and Jordan, 2001) that allows us to sample from a distribution similar to the training data. This can be accomplished by specifying a joint distribution over all the dimensions of the input. The VAE approach is an extension of autoencoder where instead of forcing encoders to create a unique encoding, we force the encoder to generate latent feature representations that roughly follow a standard normal distribution over encodings. The decoder will then sample a latent feature vector from this probability distribution and try to reconstruct the original input. In practice, there is a trade-off between the accuracy of the network and the proximity of its latent variables to the standard Normal distribution. For evaluation of neural network performance, we sum up two distinct losses- a reconstruction loss, which is a mean squared error that measures how accurately the network reconstructs the input and a latent loss, which is KL divergence (Raiber and Kurland, 2017) that measures to what extent the latent variables diverge from a standard normal distribution. The neural network aims to minimize these losses with each iteration.

Consider a training dataset *X* which consist of '*m*' samples of continuous or discrete variables $\{x^{(1)},....,x^{(i)},....,x^{(m)}\}$ where each sample has *d* dimensions i.e. $x^{(i)} \in R^d$. Suppose that the training data is generated by a random process involving unobserved (latent) continuous random variable *z*. The data generation process consists of two steps:

(1) A value $z^{(i)}$ is generated from some prior distribution $p_{\theta^*}(z)$

(2) A value $x^{(i)}$ is generated from some conditional distribution $p_{\theta^*}(x \mid z)$

The true parameters θ^* of the generative model have to be estimated in order to generate new data. So, to represent this model, select $p_\theta(z)$ to be simple, e.g. Gaussian. However, $p_\theta(x \mid z)$ is complex as it has to be used for image generation. A neural network is the best choice for representing this complex function which is called a decoder network. For training this neural network, we try to learn these model parameters in order to maximize the marginal likelihood of training data.

$$p_\theta(x) = \int p_\theta(z) p_\theta(x \mid z) dz \qquad (4)$$

Here, $p_\theta(z)$ is Gaussian prior and $p_\theta(x \mid z)$ is decoder neural network. But the marginal likelihood $p_\theta(x)$ is intractable so we cannot evaluate $p_\theta(x)$. Also the true posterior density $p_\theta(z \mid x) = p_\theta(x \mid z) p_\theta(z) / p_\theta(x)$ is intractable due to presence of $p_\theta(x)$ in it. These intractabilities appear due to complex likelihood function $p_\theta(x \mid z)$ which is represented as a neural network with non-linear hidden layer. To solve these intractibilities problems, an additional encoder network $q_\phi(z \mid x)$ (also called variational posterior) is defined that approximates the true intractable posterior $p_\theta(z \mid x)$. To know how well the variational posterior approximates the true posterior, the KL divergence is used which measures the loss in information when variational posterior is used to approximate true posterior.

$$D_{KL}(q_\phi(z \mid x) \,\|\, p_\theta(z \mid x)) = \mathrm{E}_{z \sim q_\phi(z|x)}[\log q_\phi(z \mid x) - \log p_\theta(x,z)] + \log p_\theta(x) \tag{5}$$

Our goal is to find the variational parameters ϕ that minimize this divergence. This allows us to derive a lower bound on the marginal likelihood that is tractable and can be optimized. The optimal variational posterior is therefore

$$q_\phi^*(z \mid x) = \arg\min_\phi D_{KL}(q_\phi(z \mid x) \,\|\, p_\theta(z \mid x)) \tag{6}$$

However it is intractable due to the presence of $p_\theta(x)$ in KL divergence. As KL divergence is always ≥ 0, we get

$$\log p_\theta(x) \geq L(\theta,\phi) = \mathrm{E}_{z \sim q_\phi(z|x)}(\log p_\theta(\mathrm{x},\mathrm{z}) - \log q_\phi(z \mid x)) \tag{7}$$

where *L (θ, ϕ)* is the variational lower bound on the marginal likelihood which can also be written as:

$$L(\theta,\phi) = \mathrm{E}_{z \sim q_\phi(z|x)}[\log p_\theta(x \mid z)] - D_{KL}(q_\phi(z \mid x) \,\|\, p_\theta(z)) \tag{8}$$

So, minimizing the intractable KL divergence is equivalent to maximizing the variational lower bound on the marginal likelihood that is computationally tractable and is our objective function. The first part of the objective function (denoted by equation 9) represents the quality of reconstruction and the second term regularizes the latent space towards the true posterior. To optimize the variational lower bound, a Stochastic Gradient Variation Bayes (SVGB) estimator is formulated that includes a reparameterization trick to allow the SVGB estimator to be differentiable (Kingma and Welling, 2013). A basic variational encoder network architecture is shown in figure 2.

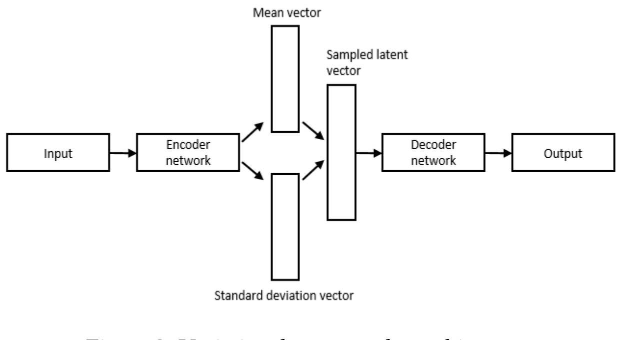

Figure 2. Variational autoencoder architecture

3. CONVOLUTION NEURAL NETWORK AND MULTINOMIAL LOGISTIC REGRESSION BACKGROUND

3.1 Convolution neural network

Convolution neural network (CNN) (Lawrence et al., 1997) is composed of alternate convolutional and max-pooling layers optionally followed by one or more fully connected layers as in standard multilayer neural network. First, the convolution layer extract features through filtering process resulting in a feature map. The feature map generated is sub-sampled by the pooling layer to generate more general and abstract features. In the final stages, various fully connected layers are utilized to generate deep features.

The convolution layer takes an image *X* of dimension *m* x *n* x *c* as input where *m* represents height, *n* represents width and *c* represents number of spectral channels in an image. For an RGB image, the value of *c* is 3 whereas for HSI, it is in order of hundred. The convolution layer consists of *k* filters or kernels denoted as *W* of size *p* x *p* x *q* such that *p* is smaller than image dimension (both *m* and *n*) and *q* is equal to the number of channels. Each kernel is convolved with the input to produce *k* features maps of dimension (*m-p*+1) x (*n-p*+1). An additive bias denoted as *b* and a non-linear activation function denoted as *f*(.) are then applied to every feature map. The output of convolution layer can be calculated as:

$$a_j = \sum_{i=1}^{c} f(x_i \otimes w_j + b_j) \quad \forall j \in [1,k] \tag{9}$$

where w_j and b_j denotes the j^{th} component of *W* and *b* respectively, the operator ⊗ denotes convolution operation, x_i denotes the i^{th} feature map of X and a_j denotes the j^{th} output of convolution layer (Song et al., 2018).

Each feature map is then sub-sampled with either max or mean pooling over *r* x *r* contiguous regions where *r* typically ranges from two to five. Finally, after a series of successive convolution and pooling operations, the resultant features are combined into one dimensional feature vector which is passed to the fully connected layers and then to the softmax layer (multinomial logistic regression) for classification. CNN architecture for image classification is shown in figure 3.

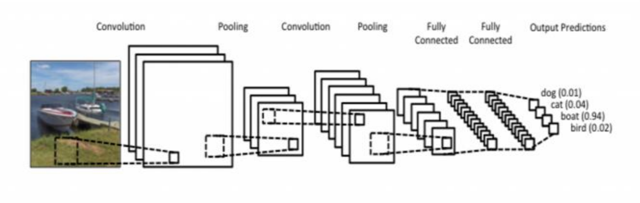

Figure 3. Convolution neural network (Gandhi, 2018)

3.2 Multinomial logistic regression

Multinomial logistic regression also called softmax regression is a generalization of logistic regression to classification problems where the output classes are more than two. It takes the numeric features of the dataset as input. For categorical features, suitable technique has to be implemented for its conversion to numerical features.

Consider a training set $\{(x^{(1)}, y^{(1)}), ..., (x^{(m)}, y^{(m)})\}$ of *m* training samples where the input features are $x^{(i)} \in \mathrm{R}^n$ and the class labels are $y^{(i)} \in \{1, 2, ...c\}$ where *c* denotes number of classes. For a given test input *x*, the hypothesis estimates $P(y = c \mid x^{(i)})$ for each *c* value i.e. the probability of the class label taking each of the *c* possible class outcomes. The hypothesis thus gives a *c* dimensional vector containing the *c* estimated probabilities (which sums up to one) as output. The hypothesis $h_\theta(x^{(i)})$ takes the form (Vryniotis, 2013)

$$h_\theta(x^{(i)}) = \begin{bmatrix} P(y=1\mid x^{(i)};\theta) \\ P(y=2\mid x^{(i)};\theta) \\ . \\ . \\ P(y=c\mid x^{(i)};\theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^{C}\exp(\theta^{(j)^T}x^{(i)})}\begin{bmatrix} \exp(\theta^{(1)^T}x^{(i)}) \\ \exp(\theta^{(2)^T}x^{(i)}) \\ . \\ . \\ \exp(\theta^{(c)^T}x^{(i)}) \end{bmatrix} \quad (10)$$

where $\theta^{(1)},\theta^{(2)},....\theta^{(c)} \in R^n$ are the model parameters and $\theta \in R^{(n\times c)}$ denotes all the parameters collectively by concatenating $\theta^{(1)},\theta^{(2)},....\theta^{(c)}$ in a column wise manner.

The Multinomial logistic regression model is formally given by (Böhning, 1992)

$$P(y^{(i)}=c\mid x^{(i)};\theta) = \frac{\exp(\theta^{(c)^T}x^{(i)})}{\sum_{j=1}^{c}\exp(\theta^{(j)^T}x^{(i)})} \quad (11)$$

And the cost function is defined as:

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{c}1\{y^{(i)}=c\}\log P(y^{(i)}=c\mid x^{(i)};\theta)\right]$$

$$= -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{c}1\{y^{(i)}=c\}\log\frac{\exp(\theta^{(k)^T}x^{(i)})}{\sum_{j=1}^{c}\exp(\theta^{(k)^T}x^{(i)})}\right] \quad (12)$$

where 1{.} is the indicator function such that 1{a true statement} = 1 and 1{a false statement} = 0 (Vryniotis, 2013)

4. METHODOLOGY

The proposed spectral-spatial framework can be divided into 3 steps as indicated through flowchart in figure 4.

1) Extraction of spectral features using variational autoencoder described in section 2.
2) Extraction of spatial features using Convolution neural network described in section 4.
3) Stacking of spectral and spatial features to form a joint feature followed by softmax regression for classification.

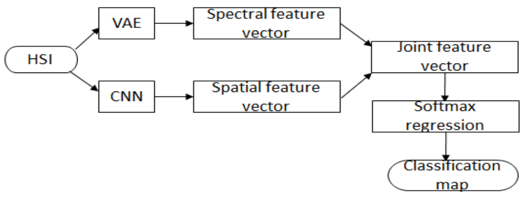

Figure 4. Flowchart for proposed framework

4.1 Spectral feature extraction using VAE

The first step is an unsupervised pre-training step using VAE for extraction of spectral features. The VAE model consist of 3 hidden encoder layers and 3 hidden decoder layers. The hidden layer neurons are 150, 100, 60 respectively. The framework for spectral feature extraction is shown in fig-7. The network is trained using exponential linear unit (ELU) (Clevert et al., 2016) activation function and using cross entropy loss function. After training, the extracted spectral features are obtained from the last hidden encoder layer.

4.2 Spatial feature extraction using CNN

The steps involved in extraction and processing of hyperspectral data are as follows:

1) PCA is performed on whole image data and PCs are obtained by setting 99.9 percent variance preserving criteria.
2) The image obtained after PCA and ground reference image is then padded with size 15 at the top, bottom, left and right such that the last element along a direction is replicated along the whole padding area in that direction.
3) A fixed window is selected centred on the non-zero ground truth pixels to generate training samples (patches). The patches will be sub-images having ground reference information corresponding to the centred ground truth pixel. (Zhao and Du, 2016)

The CNN model consist of two alternating convolution and pooling layers followed by three fully connected layers. The first convolution layer has 30 kernels of size *5* x *5* with a stride of 2 and the second convolution layer has 30 kernels of size *3* x *3* with a stride of 2. The number of neurons in fully connected layers are 1000, 400 and 60 respectively.

Suppose there are *M* training patches selected randomly from the input image and T_i denotes a training sample where $i \in [1,2,...M]$ whereas l_i denotes the class label corresponding to training patch T_i. The training objective of CNN is to learn the filters *W* and the bias parameter *b* by minimizing the cross entropy loss (Nasr et al., 2002) function using back-propagation algorithm.

$$\min J = -\sum_{i=1}^{M} l_i\log(y_i) + (1-l_i)\log(1-l_i) \quad (13)$$

where y_i = predicted class label for training patch T_i.

In order to avoid overfitting in the model, drop out regularisation (Srivastava et al., 2014) is used with 70% retain probability. After training, the extracted spatial features are obtained as an output of the last fully connected layer.

4.3 Stacking of spectral and spatial feature vectors

The spectral and spatial feature vectors obtained are stacked together to generate the joint feature vector which is passed as an input to the softmax layer for calculation of conditional probabilities for each class. For a joint feature vector *z*, the probability distribution is calculated (Song et al., 2018).

$$p_i = \frac{e^{z_i}}{\sum_{i=1}^{C}e^{z_i}}, \quad i = 1,2,....,\text{C for C classes.} \quad (14)$$

5. RESULTS

5.1 Dataset description

For various investigations, two hyperspectral datasets with different themes are utilized to validate the proposed spectral-spatial feature extraction based classification method. The first dataset is a mixed vegetation site acquired by the AVIRIS sensor over Kennedy Space Centre (KSC), Florida containing 13 land cover classes in 224 spectral bands of 10 nm width with a spatial resolution of 18 m. After removal of water absorption and low signal to noise ratio (SNR) bands, 176 bands are used for classification. The dataset is a 512 × 614 pixels image, in which ground reference information of 5211 pixels is available. The second dataset is an urban site imaged by the ROSIS sensor over the University of Pavia, Italy having nine land cover classes in 103 spectral bands of 4 nm width with a spatial resolution of 1.3 m. It is a 610 × 340 pixels image, in which ground reference information of 42776 pixels is available.

5.2 Classification accuracy analysis

For both hyperspectral datasets, we split the image into two sets, i.e., training, and testing data, with a split ratio 1:9 i.e., we randomly select 10% of the labelled samples as the training set, and remaining 90% for the testing sets, respectively. During training, we use the training set to learn weights and biases of each neuron and the test set is used to produce final classification results. In order to quantitatively compare and estimate the capabilities of the proposed models, overall accuracy (OA) and Kappa coefficient and z-scores are used as performance measurement (Congalton and Green, 2009). To perform statistical evaluation, we conduct five independent replications of the whole process and use the average Kappa coefficient to compare the performance among various DL based methods like SAE-LR (Chen et al., 2014) and CNN. The accuracy analysis is shown in Tables 1 and 2 for Kennedy Space Centre and Pavia University images respectively and figures 5 and 6 show the classification maps obtained by different methods. The classification performance comparison of the proposed (VAE+CNN)-LR method with the two other methods is done based on the one tailed hypothesis testing, the results of which are shown in Table 3. From one tailed test with 95% confidence interval, it can be concluded that the proposed method performs significantly better than the two other DL methods.

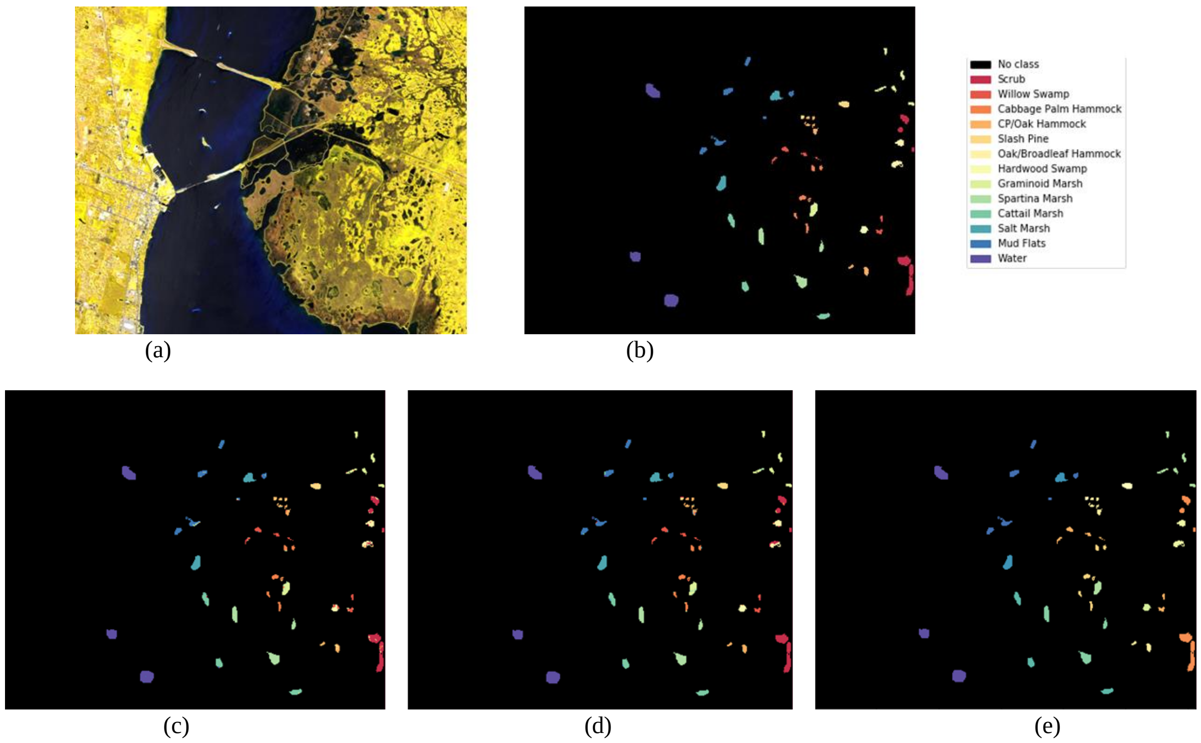

Figure 5. Best classification results of the Kennedy Space Centre scene. (a) FCC. (b) Reference map (c)-(e) classification results by using SAE-LR, CNN, (VAE+CNN)-LR

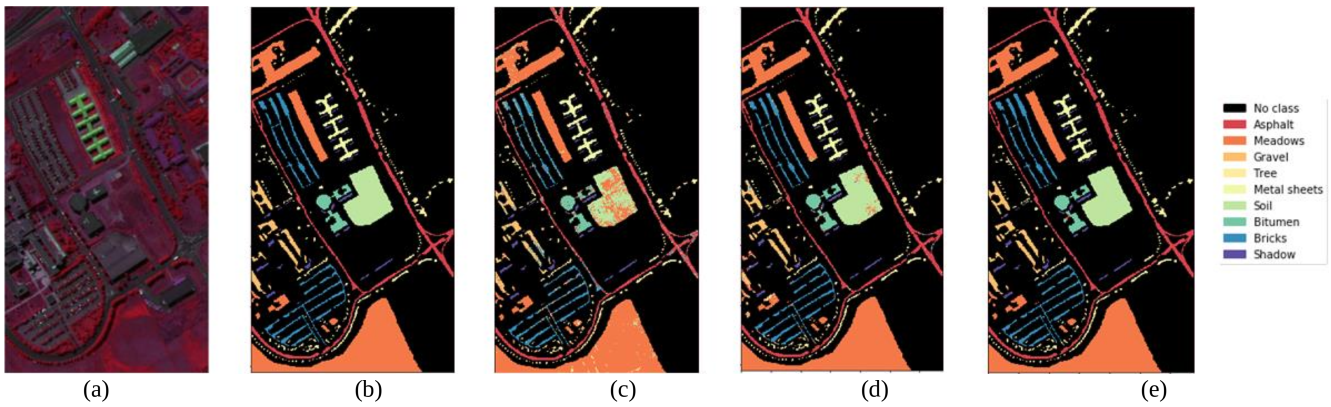

Figure 6. Best classification results of the Pavia University scene. (a) FCC (b) Ground reference map (c)-(e) classification results by using SAE-LR, CNN, (VAE+CNN)-LR

	SAE - LR		CNN		**(VAE+CNN)- LR**	
	Producer's Kappa	User's Kappa	Producer's Kappa	User's Kappa	Producer's Kappa	User's Kappa
Scrub	0.9121 ± 0.0115	0.9151 ± 0.0114	0.9679 ± 0.0074	0.8990 ± 0.0121	0.9962 ± 0.0021	0.9945 ± 0.0027
Willow swamp	0.7669 ± 0.0271	0.8933 ± 0.0214	0.9498 ± 0.0154	0.9265 ± 0.0182	0.9989 ± 0.0010	0.9920 ± 0.0053
CP hammock	0.9089 ± 0.0193	0.9047 ± 0.0197	0.9170 ± 0.0186	0.9386 ± 0.0164	0.9990 ± 0.0009	0.9990 ± 0.0009
CP/Oak	0.7242 ± 0.0307	0.7041 ± 0.0309	0.8135 ± 0.0261	0.9274 ± 0.0186	0.9861 ± 0.0077	0.9971 ± 0.0022
Slash pine	0.6761 ± 0.0383	0.7097 ± 0.0381	0.9283 ± 0.0217	0.9557 ± 0.0176	0.9940 ± 0.0040	0.9776 ± 0.0108
Oak/Broadleaf	0.7224 ± 0.0326	0.6823 ± 0.0329	0.7442 ± 0.0313	1.0000	0.9862 ± 0.0061	0.9910 ± 0.0039
Hardwood swamp	0.8582 ± 0.0415	0.6472 ± 0.0492	0.9457 ± 0.0235	1.0000	0.9978 ± 0.0021	1.0000
Graminoid marsh	0.8755 ± 0.0171	0.9074 ± 0.0153	0.8495 ± 0.0189	0.9403 ± 0.0132	0.9937 ± 0.0041	0.9914 ± 0.0047
Spartina marsh	0.9184 ± 0.0129	0.9902 ± 0.0048	0.9732 ± 0.0079	0.8714 ± 0.0155	0.9918 ± 0.0031	0.9951 ± 0.0030
Cattail marsh	0.9906 ± 0.0053	0.9328 ± 0.0134	0.9662 ± 0.0099	0.8973 ± 0.0161	0.9981 ± 0.0010-	0.9993 ± 0.0006
Salt marsh	0.9827 ± 0.0069	0.9827 ± 0.0069	0.9970 ± 0.0029	0.9685 ± 0.0093	1.0000	1.0000
Mud flats	0.9281 ± 0.0132	0.8681 ± 0.0167	0.8690 ± 0.0166	0.9336 ± 0.0127	1.0000	1.0000
Water	0.9854 ± 0.0045-	0.9840 ± 0.0047	0.9970 ± 0.0020	0.9671 ± 0.0067	1.0000	1.0000
OA	0.9095 ± 0.0041		0.9388 ± 0.0035		**0.9966 ± 0.0008**	
Kappa	0.8993 ± 0.0046		0.9317 ± 0.0039		**0.9962 ± 0.0069**	

Table 1. Class-wise accuracies of the Kennedy Space Centre (KSC) image

Class	SAE - LR		CNN		**(VAE+CNN) - LR**	
	Producer's Kappa	User's Kappa	Producer's Kappa	User's Kappa	Producer's Kappa	User's Kappa
Asphalt	0.8633 ± 0.0048	0.8403 ± 0.0050	0.9695 ± 0.0024	0.9304 ± 0.0035	0.9967 ± 0.0007	0.9956 ± 0.0009
Meadows	0.9617 ± 0.0020	0.7986 ± 0.0038	0.9921 ± 0.0009	0.9753 ± 0.0015	0.9980 ± 0.0004	0.9992 ± 0.0002
Gravel	0.5616 ± 0.0115	0.6566 ± 0.0120	0.8117 ± 0.0091	0.8979 ± 0.0074	0.9785 ± 0.0033	0.9960 ± 0.0014
Tree	0.8401 ± 0.0071	0.9409 ± 0.0048	0.9283 ± 0.0050	0.9812 ± 0.0027	0.9957 ± 0.0012	0.9904 ± 0.0019
Metal sheets	0.9867 ± 0.0033	0.9991 ± 0.0008	0.9956 ± 0.0019	0.9939 ± 0.0022	0.9981 ± 0.0012	0.9967 ± 0.0016
Soil	0.5947 ± 0.0074	0.9086 ± 0.0055	0.9597 ± 0.0030	0.9684 ± 0.0027	0.9981 ± 0.0005	0.9992 ± 0.0003
Bitumen	0.5386 ± 0.0145	0.8006 ± 0.0143	0.9602 ± 0.0057	0.9728 ± 0.0048	0.9945 ± 0.0016	0.9964 ± 0.0012
Bricks	0.8553 ± 0.0064	0.7405 ± 0.0073	0.9584 ± 0.0036	0.9344 ± 0.0044	0.9969 ± 0.0010	0.9907 ± 0.0017
Shadow	1.0000	0.9976 ± 0.0016	0.8823 ± 0.0110	0.9841 ± 0.0045	0.9997 ± 0.0024	0.9877 ± 0.0033
OA	0.8726 ± 0.0016		0.9690 ± 0.0008		**0.9971 ± 0.0002**	
Kappa	0.8278 ± 0.0022		0.9589 ± 0.0011		**0.9962 ± 0.0003**	

Table 2. Class-wise accuracies of the University of Pavia (PU) image

Dataset	(*a*) SAE-LR		(*b*) CNN		(*c*) **(VAE+CNN)-LR**		Z_{ac}	Z_{bc}
	OA	Kappa	OA	Kappa	OA	Kappa		
KSC	0.9095 ± 0.0041	0.8993 ± 0.0046	0.9388 ± 0.0035	0.9317 ± 0.0039	**0.9966 ± 0.0008**	**0.9962 ± 0.0069**	11.680	8.138
PU	0.8726 ± 0.0016	0.8278 ± 0.0022	0.9690 ± 0.00-08	0.9589 ± 0.0011	**0.9971 ± 0.0002**	**0.9962 ± 0.0003-**	75.843	31.835

Table 3. Classification performance comparison using kappa coefficient and Z-score

6. CONCLUSION

In this paper, a novel DL-based spectral-spatial framework is presented for HSI classification. The proposed method can extract more in-depth features. On the basis of z-score, the results on two hyperspectral datasets have demonstrated the superiority of the proposed method over other widely used DL methods like SAE-LR and CNN. For the future works, we propose to investigate more DL-based methods for spectral feature extraction methods and classification performance analysis both in terms of accuracy and time for hyperspectral images of different themes.

ACKNOWLEDGEMENTS

This work is supported by the Project no. STC-CE-2018038 from the Indian Space Research Organisation (ISRO), Government of India.

REFERENCES

Amari, S., 1993. Backpropagation and stochastic gradient descent method. *Neurocomputing* 5, 185–196.

Bachmann, C.M., Ainsworth, T.L., Fusina, R.A., 2005. Exploiting manifold geometry in hyperspectral imagery. *IEEE Trans. Geosci. Remote Sens.* 43, 441–454.

Bandos, T. V, Bruzzone, L., Member, S., Camps-valls, G., Member, S., 2009. Classification of Hyperspectral Images With Regularized Linear Discriminant Analysis. *IEEE Trans. Geosci. Remote Sens.* 47, 862–873.

Böhning, D., 1992. Multinomial logistic regression. *Ann. Inst. Stat. Math*. 44, 197–200.

Chen, Y., Jiang, H., Li, C., Jia, X., Member, S., 2016. Deep Feature Extraction and Classification of Hyperspectral Images Based on Convolutional Neural Networks. *IEE Trans. Geosci. Remote Sens.* 54, 1–20.

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

Clevert, D.-A., Unterthiner, T., Hochreiter, S., 2016. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs), in: *International Conference on Learning Representations.* pp. 1–14.

Congalton, R. G., and Green, K., 2009. *Assessing the Accuracy of Remotely Sensed Data: Principles and Practices,* 2nd ed. CRC Press/Taylor & Francis, Boca Raton.

Donoho, D.L.,2000. High-Dimensional Data Analysis : The Curses and Blessings of Dimensionality. *Aide-Memoire of a Lecture at AMS Conference on Math Challenges of the 21st Century*

Du, Q., Member, S., Yang, H., Member, S., 2008. Similarity Based Unsupervised Band Selection for Hyperspectral Image Analysis. *IEEE Geosci. Remote Sens. Lett.* 5, 564–568.

Gandhi, R., 2018. Build your own Convolution Neural network in 5 mins https://towardsdatascience.com/build-your-own-convolution-neural-network-in-5-mins-4217c2cf964f (20 September 2018).

Ghamisi, P., Plaza, J., Chen, Y., Li, J., Plaza, A.J., 2017. Advanced Spectral Classifiers for Hyperspectral Images: A review. *IEEE Geosci. Remote Sens. Mag*. 5, 8–32.

Gualiteri, A.J. and Cromp R.F. (1998). Support Vector Machine for hyperspectral remote sensing classification. Proc. *SPIE*. 3584, pp. 221-232

Han, T., Goodenough, D.G., 2008. Investigation of Nonlinearity in Hyperspectral Imagery Using Surrogate Data Methods. *IEEE Trans. Geosci. Remote Sens*. 46, 2840–2847.

Hu, W., Huang, Y., Wei, L., Zhang, F., Li, H., 2015. Deep Convolutional Neural Networks for Hyperspectral Image Classification. *J. Sensors* 2015, 1–12.

Hughes, G., 1968. On the Mean Accuracy of Statistical Pattern Recognizers. *IEEE Trans. Inf. Theory,* vol. IT-14, no. 1, pp. 55–63.

Kingma, D., Welling, M., 2013. Auto-Encoding Variational Bayes arXiv : 1312 . 6114v10 [stat . ML] 1 May 2014, In: *2nd International Conference on Learning Representations*. pp. 1–14.

Landgrebe, D. A., 2003. *Signal theory methods in multispectral remote sensing*. NJ: Wiley.

Lawrence, S., Giles, C.L., Member, S., Tsoi, A.C., Member, S., Back, A.D., 1997. Face Recognition : A Convolutional Neural-Network Approach. *IEEE Trans. Neural Network*. 8, 98–113.

Melgani, F., Bruzzone, L., 2004. Classification of Hyperspectral Remote Sensing. *IEEE Trans. Geosci. Remote Sens*. 42, 1778–1790.

Mou, L., Ghamisi, P., Zhu, X.X., 2017. Deep Recurrent Neural Networks for Hyperspectral Image Classification. *IEEE Trans. Geosci. Remote Sens.* 55, 3639–3655.

Nasr, G., Badr, E., Joun, C., 2002. Cross Entropy Error Function in Neural Networks: Forecasting Gasoline Demand. In: *FLAIRS Conference.* pp. 381–384.

Ng, A., Jordan, M., 2001. On Discriminative vs. Generative classifiers: A comparison of logistic regression and naive Bayes. In: *Neural Information Processing Systems*. pp. 605–610.

Raiber, F., Kurland, O., 2017. Kullback-Leibler Divergence Revisited. In: *Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval, ICTIR*. pp. 117–124.

Rodarmel, C., Shan, J., 2002. Principal Component Analysis for Hyperspectral Image Classification. *Surveying Land Inform. Sci.* 62, no. 2, pp. 115-122.

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.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2014. Dropout : A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn.* Res. 15, 1929–1958.

Villa, A. S., Benediktsson, J.A., Chanussot, J., 2011. Hyperspectral Image Classification With Independent Component Discriminant Analysis. *IEEE Trans. Geosci. Remote Sens.* 49, 4865–4876.

Vryniotis, V., 2013. Machine Learning Tutorial: The Multinomial Logistic Regression (Softmax Regression) http://blog.datumbox.com/machine-learning-tutorial-the-multinomial-logistic-regression-softmax-regression/ (10 September, 2018).

Yang, H., Member, S., Du, Q., Member, S., Su, H., Sheng, Y., 2011. An Efficient Method for Supervised Hyperspectral Band Selection. *IEEE Geosci. Remote Sens. Lett.* 8, 138–142.

Zhao, W., Du, S., 2016. Spectral-Spatial Feature Extraction for Hyperspectral Image Classification: A Dimension Reduction and Deep Learning Approach. *IEEE Trans. Geosci. Remote Sens.* 54, 4544–4554.