2020 IEEE International Symposium on Sustainable Energy, Signal Processing and Cyber Security (iSSSC) | 978-1-7281-8880-5/20/$31.00 ©2020 IEEE | DOI: 10.1109/iSSSC50941.2020.9358892

Hyperspectral Image Classification using Trilateral Filter and Deep Learning

Vedant Gupta*
DA-IICT
Gandhinagar, India
201601145@daiict.ac.in

Srikumar Sastry
DA-IICT
Gandhinagar, India
201601016@daiict.ac.in

Suman K. Mitra
DA-IICT
Gandhinagar, India
suman.mitra@daiict.ac.in

***Abstract*—The high spectral variability and resolution make hyperspectral image classification difficult. Most of the previous researches focused on solving its high dimensionality problem; very few types of researches have focused on using edge information of the spectral bands to carry out classification. This paper aims to compare two edge filtering methods - bilateral filter and trilateral filter to extract edge information of hyperspectral images and then use those features for further classification. Minimum Noise Fraction (MNF) and Principal Component Analysis (PCA) were separately used for dimensionality reduction. The images obtained from both the methods were stacked, and the two edge filters were applied to them individually. The final feature images were classified using Convolutional Neural Network (CNN). The proposed methodology was applied to the Indian Pines dataset. The results obtained are satisfactory and have been compared to state of the art (SOTA) techniques.**

***Index Terms*—Deep Learning, Minimum Noise Fraction, Hyperspectral Image Classification, Machine Learning, Principal Component Analysis, Spatial Features**

I. Introduction

Hyperspectral imaging (HSI) deals with the imagery of narrow spectral bands over a continuous spectral range. Over the past decade, HSI has been a rapidly developing field. Due to hyperspectral images discriminating ability, it has found applications in a variety of areas, such as agriculture, eye care, food processing, surveillance, chemical imaging, environment management, and land cover mapping [26]. Unlike color images with 3 bands, generally, hyperspectral images have more than 100 bands that capture both the spectral and spatial information of different objects in the image. A pixel in a hyperspectral image is a high dimensional vector where the spectral reflectance of the captured image at a particular wavelength is stored. Since HSI can detect subtle spectral differences, the classification of hyperspectral images is an important task, where the aim is to classify each pixel vector in the image into one of the various classes present in the image; as a result, pixel classification in hyperspectral images has attracted a lot of interest, [10], [14].

Most of the early methods used machine learning algorithms with hand-engineered features like Fang et. al [8] who encoded the relationship between different spectral bands using local covariance matrices and used these matrices to train a support vector machine (SVM). [16] also used SVM. Moreover, [9], [7] use dictionary-based sparse representation, and [1] uses hand-engineered features with linear discriminant analysis (LDA). However, In recent years neural networks (NN) are gaining a lot of popularity in image classification tasks, especially the convolutional neural network (CNN) [12]. This popularity stems from their ability to process spectral data and generate self-learned features that outperform models using hand generated features.

CNN is one of the most effective classification algorithm used for image classification. Over the years, it has produced many promising results in tasks requiring visual information processing like face anti-spoofing [18], image classification [27], object detection [19], etc. CNN was proposed by LeCun et. al, [13] in 1998 for the purpose of handwritten digit recognition. Yann LeCun's LeNet-5 consisted of two of each layer, i.e., convolutional layer, pooling layer and fully connected layer. Most of the CNN architectures that followed were based on this, producing state of the art results in many fields.

Lately, due to the gaining popularity of deep learning, many researchers have applied deep learning techniques on hyperspectral images, producing SOTA results [23]. Chen et. al [6] used 1D, 2D and 3D CNN for hyperspectral image classification. Santara et. al [20] proposed an architecture of neural network for learning band adaptive spatial and spectral features. Song et. al. [21] proposed a deep feature fusion network to tackle the problem of decreasing performance of CNN due to the increasing resolution of hyperspectral images. Cao et al. combined markov fields with deep learning [3]. Zhang et. al [26] proposes a model to preserve edge information in the image while simultaneously classifying the HSI image using CNN. Even though all these methods obtain good accuracy, some problems do persist in these methods. Most of the above methods ignore the information present in the edges and lose this information while preprocessing. This information as shown by zhang et. al [26] is essential for classification and affects the result drastically.

This paper tries to improve over the past researches done on preserving edge information for classification. The model proposed here achieves 98% of accuracy, which is 1.5% more than [26]. Since in hyperspectral images, both spatial and spectral features affect the class label prediction of a pixel, the paper suggests using CNN, which tries to take into account both these features. This paper proposes a CNN architecture which is based on Yann Lecun's LeNet-5 and uses novel preprocessing techniques to preserve edge information to be

978-1-7281-8880-5/20/$31.00 ©2020 IEEE

used by the classifier. This helps in increasing the useful information, which significantly improves accuracy.

The remainder of the paper is organized as follows: Section II, talks about the preprocessing techniques applied and the dataset used in this paper. It talks in detail about bilateral filter [24], PCA [15], MNF [15] and trilateral filter [5]. Section III, describes in detail the model used and the methodology. Section IV, highlights the result obtained from this approach and we conclude the paper in section V.

II. Data Set and Preprocessing Techniques

This section talks in detail about the dataset and the preprocessing techniques employed in this paper. During the first step of preprocessing, Minimum Noise Fraction (MNF) and Principal Component Analysis (PCA) are applied to the Indian Pines dataset separately. From the output of PCA, the first five principal components and from the output of MNF, all the bands with signal to noise ratio (SNR) greater than zero are selected. In the next and final step of preprocessing, the trilateral filter is applied to both these sets of images. The output is concatenated and used for training the model proposed in section III. The following subsections also talk about bilateral filtering and its drawbacks and how trilateral filters overcome these drawbacks.

A. Description of Data Set

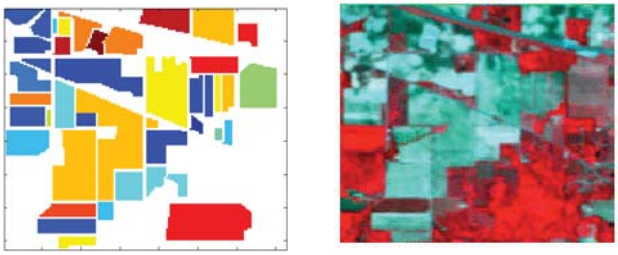
To verify the effectiveness of the model proposed in section III, we used the Indian Pines data set [2]. AVIRIS sensor was used to capture the spectral reflectance of the Indian Pines test site in North-western Indiana. The data set consists of 224 spectral reflectance bands having wavelength in the range of $0.4 - 2.5 \times 10^{-6}$ meters. After removing noise using filters described in the following subsections, only 200 of these bands remain. Each band in the data set contains 145×145 pixels. The scene in the data set has $1/3^{rd}$ forest and $2/3^{rd}$ agricultural land. Furthermore, the scene consists of crops like corn, soybeans, etc. It also has two major highways, a railway line, some buildings and roads. The dataset contains 16 classes and is available on $Purdue\ Universities\ multispec\ website$. The ground truth and false color image of the Indian pines data set are shown in figure I. Table I, shows the number of samples (pixels) of respective classes used for training and testing; the class imbalance is clearly visible in Table I. To solve this problem, in this paper, we fixed the number of training samples to 1473 and over-sampled the classes with fewer pixels by randomly copying its pixels from the train set without touching the test set.

B. Principal Component Analysis

The technique, Principal Component Analysis (PCA) is a transformation operation that converts a given correlated data into uncorrelated data. Correlated data always represents redundancy. PCA can rearrange the data into decreasing order of information. As a result, PCA can also be used for dimensionality reduction without losing much information. General equation of PCA is $Y = AX'$. Where Y is the transformed

TABLE I
NUMBER OF SAMPLES OF EACH CLASS IN TRAINING AND TESTING SET

Class	Total samples	Training samples	testing samples
Oats	20	12	8
Corn	237	142	95
Soybean	593	356	237
Corn-mintill	830	498	332
Soybean-mintill	2455	1473	982
Alfafa	46	28	18
Grass-mowed	28	17	11
trees	730	438	292
Grass-pasture	483	290	193
Corn-notill	1428	857	571
Soybean-notill	972	583	389
Wheat	205	123	82
Building-Drives	386	231	155
Woods	1265	759	506
Stone-Steel	93	56	37
Hay-windrowed	478	287	191



(a) (b)

Fig. 1. Ground truth and false color images of Indian pines data set. (a) Ground truth; (b) False color image.

uncorrelated data, X' is the original data with centered mean, and A is the transformation matrix. Let C_x and C_y be the covariance matrices of X' and Y then C_y can be written as $C_y = AC_xA^T$. We can diagonalize C_x as $C_x = U \sum U^{-1}$ where, U is the matrix of eigenvectors and $\sum$ is the eigenvalue matrix. Next, we have $A = U^{-1}$ we can write the first equation as $Y = U^{-1}X'$. To get back X from this, we can do the following:

$$UY = UU^{-1}X => X_r = UY \tag{1}$$

here, the X_r generated will not be the same as X but will have some error. To reduce the dimension of X using PCA, we can simply select the first r eigenvectors in U, where r is less than the size of U. In the case of the Indian Pines dataset, we observed using reconstruction error that 96% of the data is stored in the first five eigenvectors, so we have taken the first five principal components.

C. Minimum Noise Fraction

Minimum Noise Fraction(MNF) is a linear transformation operation which sorts the hyperspectral image bands in descending order of signal to noise ratio. MNF is a combination of two PCA rotations. In the first rotation of MNF noise covariance matrix is used to decorrelate and reconstruct the noise in the data. The decorrelated noise is called white noise, and the process is called noise whitening since it now

has zero inter-band correlation and unit variance. The noise covariance matrix used in the first rotation is calculated using shift difference at all the image pixels.

Since the noise covariance matrix(C_N) is a symmetric matrix, the eigenvectors U and V generated by the PCA are equal. The PCA decomposition of C_N becomes UD_NU^T, where D_N is a diagonal matrix with eigenvalues and U is an orthogonal matrix of eigenvectors. The following transformation $P = UD_N^{-\frac{1}{2}}$ is applied to convert the noise in the data to white noise with zero inter-band correlation and unit variance. $Y = PX$ is the formula through which image data X is mapped into the new space where noise follows the properties of white noise.

The second rotation of PCA is applied on Y. Let C_D be the covariance matrix of image X and C_M be the transformed covariance matrix, then C_M is given by $C_M = P^T C_D P$. PCA diagonalize this C_M matrix into D_M matrix which is given by $D_M = V^T C_M V$. Here, D_M is the diagonal matrix with eigenvalues of C_M and V is a matrix of eigenvector of C_M. The MNF transformation matrix is obtained as $T_{MNF} = PV$.

D. Bilateral Filter

Bilateral filter was introduced by C. Tomasi and R. Manduchi [24]. It is a simple non-iterative filter used for edge-preserving smoothing and is very similar to the Gaussian filter. It also uses a convolutional kernel and the value at a pixel after filtering depends upon the weighted sum of corresponding neighborhood pixels. The only difference between a bilateral filter and a Gaussian filter is that both employ different kernels. This difference in kernel functions helps bilateral filter to preserve edge-information in an image.

In a Gaussian filter the kernel weights are dependent only on spatial distance between the pixels. The points closer to the target pixels have higher weight and vise versa. The formula for Gaussian filtering is $G_i = 1/D \sum_i \sum_j f_j dis(i,j)$ and $D = \sum_i \sum_j dis(i,j)$ where, f_j is the value of Image at pixel j in the neighborhood of pixel i and $dis(i,j)$ is the kernel weight at pixel j based on the spatial distance between pixel i and j. This approach by Gaussian filter fails to preserve edge information. To solve this problem bilateral filter adds a new component to Gaussian filter, which helps in preserving edges. The formula for the same is $E_i = 1/S \sum_i \sum_j f_j s(f_i, f_j)$ and $S = \sum_i \sum_j s(f_i, f_j)$ where, $s(f_i, f_j)$ is the Gaussian weight based on the similarity between the pixels. When these two components are combined a bilateral filter is formed whose equation is given by:

$$O_i = 1/K_i \sum_i \sum_j I_j dis(i,j) s(I_i, I_j) \tag{2}$$

$$K_i = \sum_i \sum_j dis(i,j) s(I_i, I_j) \tag{3}$$

where, $dis(i,j)$ and $s(I_i, I_j)$ are given by,

$$dis(i,j) = e^{-1/2(||i-j||^2/\sigma_d)^2} \tag{4}$$

$$s(I_i, I_j) = e^{-1/2(||I_i - I_j||^2/\sigma_s)^2} \tag{5}$$

σ_d and σ_s are hyperparameters, which can be changed to change the kernel size and amount of filtering.

Even though bilateral filtering has found many applications in computer vision field [11] namely denoising, contrast management, depth reconstruction, data fusion and 3D reconstruction, it still suffers from many drawbacks. The three main drawbacks of bilateral filters are [5]:

- Bilateral filters have problems preserving sharp changes in gradients. They cause a blurring effect on ramp-edges or ridge-like features.
- Bilateral filter generally performs poorly on curvature and high gradient areas in an image as most of the nearby region's pixels are outliers that miss the kernel window.
- Bilateral filter with a wide window may include pixels on either side of high gradient region, which may cause irregular smoothing.

To solve these problems and improve the amount of edge-information preserved in the filtered image, we propose the use of a trilateral filter, which is described in the next subsection.

E. Trilateral Filter

The trilateral filter was introduced by W.C.K Wong et al. [25]. The trilateral filter uses the principles of bilateral filter. Novel contributions to computer vision made by trilateral filter are:

- Tilting: The kernel window in a trilateral filter is tilted by a gradient vector G_θ which is calculated by applying bilateral filter to image data, this helps in capturing high gradient regions in the image.
- Adaptive Region Growing: In a trilateral filter, the size of the local neighborhood can be controlled by a parameter R to smooth only regions having similar gradients.
- One parameter: All the different parameters in a trilateral filter can be derived from a single parameter provided by the user.

Tilting in a trilateral filter helps solve the problem of poor filtering of high gradient regions faced by bilateral filters. The tilting angle G_θ of a trilateral filter is calculated by applying a bilateral filter to the image data, since $G_\theta(x)$ at target pixel x should average strongly related pixel in the neighborhood and ignore dissimilar pixels, we use bilateral filters as they are good at this. The formula is given by:

$$G_\theta(x) = 1/K_\theta \sum_x \sum_y I_y d(x,y) s(I_x, I_y) \tag{6}$$

$$K_\theta = \sum_x \sum_y d(x,y) s(I_x, I_y) \tag{7}$$

Tilting the kernel in a trilateral filter causes the functions $d()$ and $s()$ to become non-orthogonal. To revert them back to orthogonal we instead of measuring the similarity between the target pixel and the pixel in the neighborhood, we measure the similarity of the neighborhood pixel with a plane passing through the target pixel. The value of each pixel at this plane is calculated by $P(x,y) = I(x) + G_\theta.(||x-y||)$. Where, $I(x)$

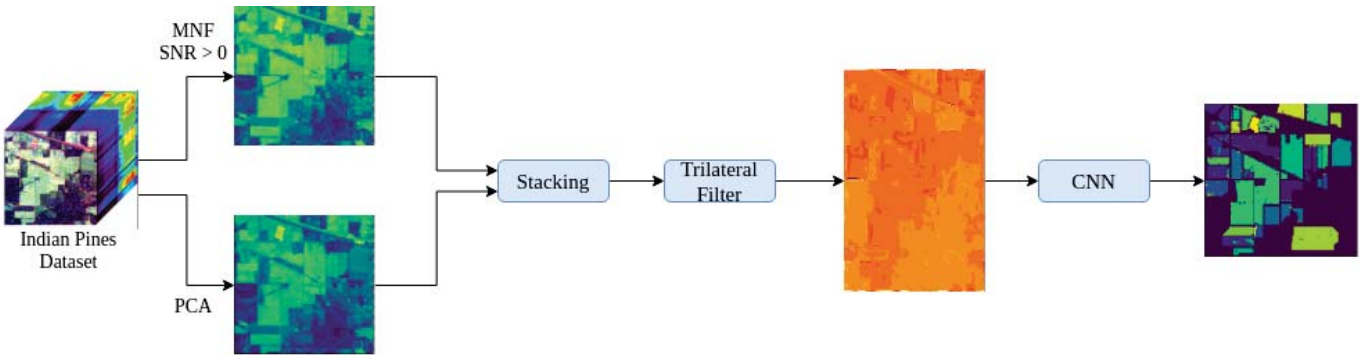


Fig. 2. The schematic diagram of the proposed classification model with the final output

at a target pixel x represents its intensity, G_θ is the tilting angle and $||x-y||$ is the spatial distance between x and y.

The output of a trilateral filter is calculated by subtracting the value P from the neighborhood of the target pixel and applying bilateral filter on that image. It is given by the following formula:

$$I_\delta = I_{in}(x) - P(x,\Delta) \tag{8}$$

$$L(x) = 1/K_\delta \int_\infty^\infty I_\delta(x,\Delta)c(\Delta)s(I_\delta(x,\Delta))f_\theta(x,\Delta)d\Delta \tag{9}$$

$$I_{out}(x) = I_{in}(x) + L(x) \tag{10}$$

where, Δ is the spatial distance between pixel x and y and $f_\theta(x,\Delta)$ is a binary function with values in the range [0,1]. Tilting helps improve the smoothing power of trilateral filter by allowing it to smooth high gradient regions properly. Still, alone it is not enough as with only tilting trilateral filter will fail at regions having stark gradient changes.

To address this issue trilateral filters apply a threshold R on the difference between the magnitude of gradients at the target pixel and the pixel in the neighborhood to form the binary function $f_\theta(x,\Delta)$. This allows trilateral filter to limit the window to regions having similar gradient values to the target pixel, hence solving first and third problem faced by bilateral filter. The binary function $f()$ is given by:

$$f_\theta(x,\Delta) = \begin{cases} 1, & \text{if } ||G_\theta(x+\Delta) - G_\theta(x)|| < R \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

III. Proposed Model

Figure II shows the schematic diagram of the proposed classification method. The previous section discussed the preprocessing techniques used in the proposed classification method; this section talks about the architecture of CNN used. A neural network is composed of many neurons; they are the basic constituent of a neural network. A neuron functions by taking the weighted sum of all the inputs coming in it and passing this sum through a non-linear activation function.

A CNN is a multi-layer neural network formed by combining pooling layers, fully connected layers and convolutional layers. The convolutional layer in a CNN simply uses filters or kernels to generate feature maps which can be processed further. The function of a pooling layer in a CNN is to reduce the spatial dimension of the representation produced by the convolutional layer, i.e. the feature maps. This helps in reducing the computation in the network by reducing the number of parameters.

This paper proposes a CNN architecture based on Yann Lecun's LeNet-5. The proposed CNN architecture have 2 convolutional layers with 5x5 kernel, 2 max-pooling layer with 2x2 kernel and 2 fully connected layers as in LeNet-5's architecture. After experimenting on the size of the training samples, the size of 25x25 is used in this paper. Figure III, shows the architecture of CNN used in this paper. When the dimension of the input patch is 25x25, the first convolutional layer generates 20 feature maps of dimension 21x21. After the application of max pooling, the dimension of each feature map is reduced to 10x10; these feature maps are then again passed through a convolutional layer and a max pooling layer. The output of this pooling layer is passed through 2 fully connected layers, which generate a column vector of size C+1, where C is the number of classes in the Indian Pines dataset. The CNN was trained using the ADAM optimizer. It gave an optimal performance at about 70 epochs. The loss function used was binary cross-entropy with cross-validation enabled.

IV. Results

In this paper, the kappa coefficient and the overall accuracy (OA) are used for measuring the performance of the model. For the purpose of objective comparison, the proposed model is compared with CNN, K-nearest neighbor (KNN)[22], normal SVM[17], locality fisher discrimination (LFDA) method with SVM, PCA method without bilateral or trilateral filter with CNN and the model proposed by Zhang et al. [26]. Table II shows the results.

The KNN is a method for finding k nearest neighbour of a sample and putting that sample in the same class as that of the neighbours. SVM is simply applying support vector machine directly on hyperspectral data for the purpose of classification. LFDA-SVM uses LFDA for the purpose of dimensionality reduction of hyperspectral images and applies SVM on the new data. PCA-CNN is a cascade of PCA, for dimensionality

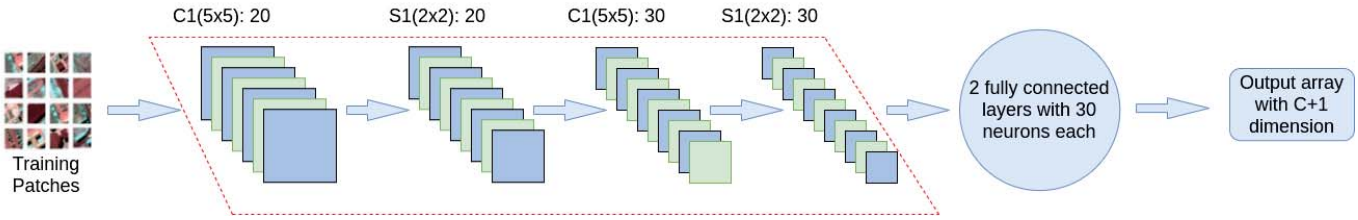


Fig. 3. The proposed CNN architecture with 2 convolutional layers with 5×5 kernels, 2 max pooling layers with 2×2 kernels and 2 fully connected layers with 30 neurons with input patches of dimension 25×25

TABLE II
KAPPA COEFFICIENT AND OA OF DIFFERENT CLASSIFIERS FOR INDIAN PINES DATASET

Models	Overall Accuracy	kappa coefficient
P-Model	98	96.89
MNFB-CNN	96.52	96.20
CNN	94.04	93.26
PCA-CNN	95.75	95.15
SVM	76.45	70.93
KNN	80.43	77.61
LFDA-SVM	80.40	74.63

reduction and CNN, for feature learning and classification. MNFB-CNN is the method proposed by Zhang et al., it uses MNF and bilateral filter in the preprocessing step and passes this preprocessed data through a CNN. P-Model is the model proposed in this paper.

From table II, it is clear that the worst performing classifier is SVM. Using only PCA with CNN results in increased accuracy of 95.75%, which is further increased by approximately 1% if the edges in the images are preserved using bilateral filter as shown in the results obtained by MNFB-CNN. The model proposed in this paper achieves the best accuracy of 98%, increasing the accuracy of MNFB-CNN by 1.5% with the help of enhanced edge preservation using trilateral filter showing the importance of edge information in hyperspectral images also showing that trilateral filters are better than bilateral filters for the purpose of preserving edges. Figure II shows the classification result of the proposed model on the Indian Pines dataset.

Next, Table III shows the results of the comparison of the proposed model in this paper with SOTA methods for hyperspectral image classification on the Indian Pines dataset.

TABLE III
COMPARISON WITH SOTA METHODS FOR CLASSIFICATION ON INDIAN PINES DATASET

Author	Model	OA	kappa
S. K. Roy et al.[23]	HybridSN	99.75	99.63
Santara et al.[20]	Bass Net	96.77	96.12
X. Cao et al. [4]	CNN	96.12	95.78
Zhang et al. [26]	MNFB-CNN	96.52	96.20
Our Model	P-Model	98	96.89

Even though HybridSN proposed by S. K. Roy et al. [23] has a much larger OA and kappa coefficient than our model, it uses a 2D-CNN after a 3D-CNN. This is a very complex model and requires a lot of training. Our model, which is comparatively very simple, produces comparable results to HybridSN and improves the accuracy of other SOTA methods by 1.5%, showing the importance of edge information in the images and how extracting it can help improve classification significantly without any added complexity.

V. CONCLUSION

In conclusion, PCA helped in removing noise from the image, it also helped in selecting bands in which majority of the information was stored. MNF helped in redistributing and removing noise from the image, while also helping in extraction of information from the image. The spatial information extracted after applying PCA and MNF is degraded to a certain degree. So, edge information was extracted using the trilateral filter. Extracting edge information is a powerful feature extraction technique as it can separate discrete objects with the help of the edge information. Extracting edge information helps in improving classification as seen by an increase of 1.5% in the OA of our model from MNFB-CNN model proposed by Zhang et al. [26]. This shows that increasing the quality of extracted edge information helps in increasing the classification accuracy of Machine Learning based models without introducing much complexities in the model.

REFERENCES

[1] T. V. Bandos, L. Bruzzone, and G. Camps-Valls. "Classification of Hyperspectral Images With Regularized Linear Discriminant Analysis". In: *IEEE Transactions on Geoscience and Remote Sensing* 47.3 (2009), pp. 862–873.

[2] Marion F. Baumgardner, Larry L. Biehl, and David A. Landgrebe. *220 Band AVIRIS Hyperspectral Image Data Set: June 12, 1992 Indian Pine Test Site 3*. Sept. 2015. DOI: doi:/10.4231/R7RX991C. URL: https://purr.purdue.edu/publications/1947/1.

[3] H. Cao and G. Venu. "Preprocessing of low-quality handwritten documents using markov random fields". In: *IIEEE Transactions on Pattern Analysis and Machine Intelligence*. 2009, pp. 1194–1194.

[4] X. Cao, F. Zhou, L. Xu, D. Meng, Z. Xu, and J. Paisley. "Hyperspectral Image Classification With Markov Random Fields and a Convolutional Neural Network". In: *IEEE Transactions on Image Processing* 27.5 (2018), pp. 2354–2367.

[5] P. Chaudhry and J. Tumblin. “The trilateral filter for high contrast images and meshes”. In: *Proceedings of Eurographics Symp. Rendering*. 2003, pp. 1–11.

[6] Y. Chen, H. Jiang, C. Li, X. Jia, and P. Ghamisi. “Deep Feature Extraction and Classification of Hyperspectral Images Based on Convolutional Neural Networks”. In: *IEEE Transactions on Geoscience and Remote Sensing* 54.10 (2016), pp. 6232–6251.

[7] Y. Chen, N. M. Nasrabadi, and T. D. Tran. “Hyperspectral Image Classification Using Dictionary-Based Sparse Representation”. In: *IEEE Transactions on Geoscience and Remote Sensing* 49.10 (2011), pp. 3973–3985.

[8] L. Fang, N. He, S. Li, A. J. Plaza, and J. Plaza. “A New Spatial–Spectral Feature Extraction Method for Hyperspectral Images Using Local Covariance Matrix Representation”. In: *IEEE Transactions on Geoscience and Remote Sensing* 56.6 (2018), pp. 3534–3546.

[9] L. Fang, S. Li, X. Kang, and J. A. Benediktsson. “Spectral–Spatial Hyperspectral Image Classification via Multiscale Adaptive Sparse Representation”. In: *IEEE Transactions on Geoscience and Remote Sensing* 52.12 (2014), pp. 7738–7749.

[10] M. Fauvel, Y. Tarabalka, J. A. Benediktsson, J. Chanussot, and J. C. Tilton. “Advances in Spectral-Spatial Classification of Hyperspectral Images”. In: *Proceedings of the IEEE* 101.3 (2013), pp. 652–675.

[11] Pierre Kornprobst, Jack Tumblin, and Frédo Durand. “Bilateral Filtering: Theory and Applications”. In: *Foundations and Trends in Computer Graphics and Vision* 4 (Jan. 2009), pp. 1–74. DOI: 10.1561/0600000020.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[13] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[14] W. Li, F. Feng, H. Li, and Q. Du. “Discriminant Analysis-Based Dimension Reduction for Hyperspectral Image Classification: A Survey of the Most Recent Advances and an Experimental Comparison of Different Techniques”. In: *IEEE Geoscience and Remote Sensing Magazine* 6.1 (2018), pp. 15–34.

[15] Guangchun Luo, Guangyi Chen, Ling Tian, Ke Qin, and Shen-En Qian. “Minimum Noise Fraction versus Principal Component Analysis as a Preprocessing Step for Hyperspectral Imagery Denoising”. In: *Canadian Journal of Remote Sensing* 42 (Mar. 2016), pp. 00–00. DOI: 10.1080/07038992.2016.1160772.

[16] F. Melgani and L. Bruzzone. “Classification of hyperspectral remote sensing images with support vector machines”. In: *IEEE Transactions on Geoscience and Remote Sensing, vol. 42, no. 8*. Aug. 2004, pp. 1778–1790.

[17] G. Mercier and M. Lennon. “Support vector machines for hyperspectral image classification with spectral-based kernels”. In: *IGARSS 2003. 2003 IEEE International Geoscience and Remote Sensing Symposium. Proceedings (IEEE Cat. No.03CH37477)*. Vol. 1. 2003, 288–290 vol.1.

[18] C. Nagpal and S. R. Dubey. “A performance evaluation of convolutional neural networks for face anti spoofing”. In: *IEEE International Joint Conference on Neural Networks (IJCNN)*. 2019.

[19] S. Ren, K. He, R. Girshick, and J. Sun. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.6 (2017), pp. 1137–1149.

[20] A. Santara, K. Mani, P. Hatwar, A. Singh, A. Garg, K. Padia, and P. Mitra. “BASS Net: Band-Adaptive Spectral-Spatial Feature Learning Neural Network for Hyperspectral Image Classification”. In: *IEEE Transactions on Geoscience and Remote Sensing, vol. 55*. June 2017, pp. 5293–5301.

[21] W. Song, S. Li, L. Fang, and T. Lu. “Hyperspectral Image Classification With Deep Feature Fusion Network”. In: *IEEE Transactions on Geoscience and Remote Sensing* 56.6 (2018), pp. 3173–3184.

[22] W. Song, S. Li, X. Kang, and K. Huang. “Hyperspectral image classification based on KNN sparse representation”. In: *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. 2016, pp. 2411–2414.

[23] Swalpa Kumar Roy, Gopal Krishna, Shiv Ram Dubey, and Bidyut B. Chaudhuri. “HybridSN: Exploring 3-D–2-D CNN Feature Hierarchy for Hyperspectral Image Classification”. In: *IEEE Geoscience and Remote Sensing Letters* 17.2 (2020), pp. 277–281.

[24] C. Tomasi and R. Manduchi. “Bilateral Filtering for Gray and Color Images”. In: *Proceedings of the 1998 IEEE International Conference on Computer Vision*.

[25] W. C. K. Wong, A. C. S. Chung, and S. C. H. Yu. “Trilateral filtering for biomedical images”. In: *2004 2nd IEEE International Symposium on Biomedical Imaging: Nano to Macro (IEEE Cat No. 04EX821)*. 2004, 820–823 Vol. 1.

[26] Dexiang Zhang, Jingzhong Kang, Lina Xun, and Yu Huang. “Hyperspectral Image Classification Using Spatial and Edge Features Based on Deep Learning”. In: *International Journal of Pattern Recognition and Artificial Intelligence* 33 (Dec. 2018). DOI: 10.1142/S0218001419540272.

[27] H. Zhang, Y. Li, and Q. Shen. “Spectral-spatial classification of hyperspectral imagery using a dual-channel convolutional neural network.” In: *Remote Sensing Letters*. 2017, pp. 438–447.