Spectral-Spatial Classification of Hyperspectral Images Using CNNs and Approximate Sparse Multinomial Logistic Regression

Sezer Kutluk
Electrical-Electronics Engineering Dept.
Istanbul University-Cerrahpasa
Istanbul, Turkey
sezer.kutluk@gmail.com

Koray Kayabol
Electronics Engineering Dept.
Gebze Technical University
Kocaeli, Turkey
koray.kayabol@gtu.edu.tr

Aydin Akan
Biomedical Engineering Dept.
Izmir Katip Celebi University
Izmir, Turkey
aydin.akan@ikc.edu.tr

***Abstract*—We propose a technique for training convolutional neural networks (CNNs) in which the convolutional layers are trained using a gradient descent based method and the classification layer is trained using a second order method called approximate sparse multinomial logistic regression (ASMLR) which also provides a spatial smoothing procedure that increases the classification accuracy for hyperspectral images. ASMLR performs well on hyperspectral images, and CNNs are known to give good results in many applications such as image classification and object recognition. Thus, the proposed technique allows us to improve the performance of CNNs by training the whole network with an end-to-end framework. This approach takes advantage of convolutional layers for spectral feature extraction, and of the softmax classification layer for feature selection with sparsity constraints, and an intrinsic learning rate adjustment mechanism. In classification, we also use a spatial smoothing method. The proposed method was evaluated on two hyperspectral images for spectral-spatial land cover classification, and the results have shown that it outperforms the CNN and the ASMLR classifiers when they are used separately.**

***Index Terms*—hyperspectral image classification, remote sensing, deep learning, convolutional neural networks, logistic regression**

I. Introduction

Convolutional neural networks (CNNs) are utilized in many fields of image analysis, computer vision, and related research and applied areas. Since CNNs yield very successful results in many fields, in recent years, they have been applied to hyperspectral image classification problems as well.

Hyperspectral image classification has been an active research topic in several areas such as remote sensing, biomedical imaging, surveillance, and food processing. Particularly in land cover classification the aim is to segment hyperspectral images obtained from aerial vehicles by pixel-wise labeling which has a lot of applications such as urban mapping, agricultural planning, and forest monitoring.

Due to the nature of the problem, small training size has always been an issue in land cover classification on hyperspectral images. The number of spectral bands is very large when compared to the number of pixels in most hyperspectral classification datasets available. Therefore Hughes phenomenon [1] can easily be encountered which makes the problem harder to solve, especially when using deep networks since they generally need huge training sets because of the large number of parameters they have.

In [2], 1-D, 2-D and 3-D deep convolutional network architectures are proposed for feature extraction from spectral bands, spatial information, and both. Moreover, in order to avoid overfitting, regularization with L_2 norm, rectified linear unit (ReLU), and dropout are used, and the training size is increased by generating virtual samples. In [3], convolution layers including deconvolution operations are utilized. Extreme learning machine is used for classification. In [4], single and double layer CNNs are tested for the classification of hyperspectral images. Each spectral vector is expanded and folded to form a matrix. Then the filters of different sizes are applied to each pixel matrix. The classification performance of CNNs are compared to support vector machine (SVM). In [5], the spectral dimensionality is reduced by a subspace learning technique and image patches are applied to a CNN for further feature learning and classification. In [6], a two-channel CNN is proposed for jointly learning spectral-spatial features. Each channel works on either spectral or spatial feature extraction. The extracted spectral and spatial features are then concatenated and given to a fully connected layer for jointly learning the spectral-spatial features. Moreover, a transfer learning scheme is employed in order to overcome the small sample size problem. In [7], 3-D convolutional filters with different kernel sizes are used for jointly learning spectral and spatial features. In this work inception module [8] and residual learning approach [9] are utilized. In [10] the hyperspectral image is segmented into superpixels and then given to a deep CNN to label each superpixel. A conditional random field (CRF) is used with a mean-field approximation algorithm formulated with Gaussian pairwise potentials as a recurrent neural network (RNN). Each iteration of the mean-field algorithm is formulated as a stack of CNN layers and the mean-field inference is considered as an RNN. In [11]

This work is supported by Scientific and Technological Research Council of Turkey (TUBITAK) under Project No. 114E535.

978-9-0827-9703-9/19/$31.00 ©2019 IEEE

the most informative spectral bands are iteratively selected using fractional order Darwinian particle swarm optimization in order to overcome the lack of training samples and the curse of dimensionality. The selected bands are given to the CNN for classification. In this work dither [12] is used for the regularization of the CNN parameters in order to avoid overfitting. In [13] a CNN is used to extract spectral-spatial features from 3-D patches of the hyperspectral image. Furthermore, with the Bayesian approach, spatial priors are placed on the labels within a Markov Random Field (MRF) model. The MRF is optimized with the α-expansion min cut method [14].

Typically CNNs are trained using first order derivative-based optimization techniques by backpropagating the loss calculated at the output of the network to the former layers. In this paper a novel convolutional neural network is proposed with a hybrid training strategy which utilizes both first and second order optimization techniques. The feature extraction layers are trained with a gradient descent based algorithm, and the softmax classification layer is trained with ASMLR [15]. ASMLR was shown to perform well on hyperspectral image classification by modeling spectral signatures of pixels and spatial smoothing by taking advantage of the neighborhood information with a Bayesian approach [15], [16].

The proposed method has the following advantages:

- second order optimization helps us avoid from hand-tuning the learning rate for the classification layer
- approximate learning helps us avoid calculating and inverting the exact Hessian matrix
- sparsifying the classification parameters works as a feature selection method
- regularizing the last layer helps us avoid overfitting and train the model with a relatively small training set
- integrated spatial smoothing by placing priors on the class labels
- end-to-end training.

The rest of the paper is organized as follows. In Section II the proposed method is presented. In Section III experiment design and results are given. The results and future work are discussed in Section IV.

II. Method

The proposed model consists of three convolutional layers where each layer is constructed by a combination of convolution, batch normalization, ReLU, and max pooling.

While training the network, in each epoch, first the classification layer is trained by ASMLR update rule, and convolutional layer parameters are updated by backpropagation and gradient descent. Spatial smoothing does not have parameters to train on the training data.

A. Convolutional Neural Networks

Convolutional neural networks are multilayer neural networks that include the convolution operation in at least one layer [17]. A typical convolutional neural network is constructed by stacking pairs of convolution layers and pooling operators as well as additional layers to model the nonlinearities and to overcome the overfitting problem. It was shown that CNNs are good at learning abstract features automatically by using convolution layers and pooling operators [18].

We use 1-D convolutional layers stacked to form the feature extraction layers and apply the spectral vectors to the network as the input data. We apply the backpropagation procedure for training the parameters of the convolutional layers with the ADAM optimization algorithm [19]. The last layer consists of a fully connected layer, where the number of units is equal to the number of classes, followed by a softmax function in order to obtain normalized probability values. Parameters of the softmax classification layer are estimated with the ASMLR [15] algorithm.

B. Approximate Sparse Multinomial Logistic Regression

ASMLR [15] is a probabilistic classifier which models the pixels as a mixture of multinomial logistic regression models and takes advantage of pixel neighborhoods. In this model a pixel vector $\mathbf{s}_n$ for $n = 1, 2, ..., N$ is assumed to be generated by one of K multinomial distributions where N is the number of pixels and K is the number of classes.

1) Model:

$$p(\mathbf{s}_n|z_{n,k} = 1, \boldsymbol{\omega}_{1:K}) = \frac{e^{\boldsymbol{\omega}_k^T \mathbf{s}_n}}{\sum_{j=1}^{K} e^{\boldsymbol{\omega}_j^T \mathbf{s}_n}} \tag{1}$$

where $\boldsymbol{\omega}_k$ is the regression coefficient vector of the k^{th} class. By defining the label vector $\mathbf{z}_n \in \{0, 1\}^K$ with the property that $\sum_{k=1}^{K} z_{n,k} = 1$, we can write (1) as follows:

$$p(\mathbf{s}_n|\mathbf{z}_n, \boldsymbol{\omega}_{1:K}) = \prod_{k=1}^{K} \left(\frac{e^{\boldsymbol{\omega}_k^T \mathbf{s}_n}}{\sum_{j=1}^{K} e^{\boldsymbol{\omega}_j^T \mathbf{s}_n}} \right)^{z_{n,k}} \tag{2}$$

We can write the joint density of $\mathbf{s}_n$ and $\mathbf{z}_n$ for all pixels as follows:

$$p(\mathbf{s}_{1:N}, \mathbf{z}_{1:N}|\boldsymbol{\omega}_{1:K}, \beta) = \left[\prod_{n=1}^{N} \prod_{k=1}^{K} p(\mathbf{s}_n|\boldsymbol{\omega}_k)^{z_{n,k}} \right] p(\mathbf{z}_{1:N}|\beta) \tag{3}$$

We obtain the sparse regression coefficients by placing Laplace priors over the coefficients:

$$p(\boldsymbol{\omega}_{1:K}|\lambda) = \prod_{k=1}^{K} \frac{\lambda}{2} e^{-\lambda ||\boldsymbol{\omega}_k||_1} \tag{4}$$

where $||\boldsymbol{\omega}_k||_1 = \sum_{l=1}^{L} |\omega_{k,l}|$ is the l_1 norm.

2) Parameter Estimation: We use the maximum a-posteriori estimate of the regression coefficients $\boldsymbol{\omega}_k$. For this purpose, we can write the log-posterior as follows:

$$\begin{aligned} L(\boldsymbol{\omega}) \quad = \quad & \sum_{n=1}^{N} \left[\sum_{k=1}^{K} z_{n,k} \boldsymbol{\omega}_k^T \mathbf{s}_n - \log \sum_{j=1}^{K} \exp(\boldsymbol{\omega}_j^T \mathbf{s}_n) \right] \\ & -\lambda \sum_{k=1}^{K} ||\boldsymbol{\omega}_k||_1 \end{aligned} \tag{5}$$

where $\boldsymbol{\omega} = [\boldsymbol{\omega}_1^T, ..., \boldsymbol{\omega}_K^T]^T$. The second order Taylor series expansion of $L(\boldsymbol{\omega})$ around $\boldsymbol{\omega}^{(t)}$ can be written as follows:

$$L(\boldsymbol{\omega}) - L(\boldsymbol{\omega}^{(t)}) = (\boldsymbol{\omega} - \boldsymbol{\omega}^{(t)})^T \mathbf{g}_L(\boldsymbol{\omega}^{(t)}) + \frac{1}{2}(\boldsymbol{\omega} - \boldsymbol{\omega}^{(t)})^T \mathbf{H}_L(\boldsymbol{\omega}^{(t)})(\boldsymbol{\omega} - \boldsymbol{\omega}^{(t)}) \quad (6)$$

where $\mathbf{H}_L(\boldsymbol{\omega}^{(t)})$ is the Hessian matrix and $\mathbf{g}_L(\boldsymbol{\omega}^{(t)})$ is the gradient vector. Hessian can be written as the sum of the Hessian matrices obtained from the log-likelihood and the log-prior as follows:

$$\mathbf{H}_L(\boldsymbol{\omega}^{(t)}) = \mathbf{H}_l(\boldsymbol{\omega}^{(t)}) + \lambda\Lambda(\boldsymbol{\omega}^{(t)}) \quad (7)$$

By maximizing (7), we can write the following:

$$\boldsymbol{\omega}^{(t+1)} = \boldsymbol{\omega}^{(t)} - (\mathbf{H}_l(\boldsymbol{\omega}^{(t)}) + \lambda\Lambda(\boldsymbol{\omega}^{(t)}))^{-1}\mathbf{g}_L(\boldsymbol{\omega}^{(t)}) \quad (8)$$

which is the update rule from the Newton's method for optimization. In [20] it was shown that Hessian from the log-likelihood is lower bounded with a constant as follows:

$$\mathbf{H}_L(\boldsymbol{\omega}) = \mathbf{H}_l(\boldsymbol{\omega}) + \lambda\Lambda(\boldsymbol{\omega}) \geq \mathbf{B} + \lambda\Lambda(\boldsymbol{\omega}) \quad (9)$$

With this lower bound the update rule can be written as follows:

$$\boldsymbol{\omega}^{(t+1)} = \boldsymbol{\omega}^{(t)} - (\mathbf{B} + \lambda\Lambda(\boldsymbol{\omega}^{(t)}))^{-1}\mathbf{g}_L(\boldsymbol{\omega}^{(t)}) \quad (10)$$

Equation (10) can be calculated component-wise as in coordinate descent algorithm in order to avoid large matrix inversion:

$$\boldsymbol{\omega}_k^{(t+1)} = \boldsymbol{\omega}_k^{(t)} - \left[\mathbf{B}_{kk} + \lambda\Lambda(\boldsymbol{\omega}_k^{(t)})\right]^{-1}\Big[g_k(\boldsymbol{\omega}_k^{(t)}) + \frac{1}{2}\sum_{j\neq k}(\mathbf{B}_{kj} + \lambda\Lambda(\boldsymbol{\omega}_j^{(t)})\mathbf{e}_j + \lambda\text{sign}(\boldsymbol{\omega}_k^{(t)})\Big] \quad (11)$$

where $\mathbf{e}_j = \boldsymbol{\omega}_j^{(t)} - \boldsymbol{\omega}_j^{(t-1)}$.

Each block of the lower bound matrix $\mathbf{B}$ can be calculated as follows:

$$\mathbf{B}_{kj} = -\frac{1}{2}(\delta_{kj} - 1/K)\mathbf{S}^T\mathbf{S} \quad (12)$$

where $\mathbf{S} = [\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_N]^T$, and δ_{kj} is the Kronecker delta function. The gradient with respect to $\boldsymbol{\omega}_k$ can be calculated as follows:

$$g_k(\boldsymbol{\omega}_k^{(t)}) = \sum_{n=1}^{N}(z_{n,k} - \pi_{n,k})\mathbf{s}_n \quad (13)$$

where

$$\pi_{n,k} = \frac{e^{\boldsymbol{\omega}_k^T\mathbf{s}_n}}{\sum_{j=1}^{K} e^{\boldsymbol{\omega}_j^T\mathbf{s}_n}} \quad (14)$$

In (11) Λ function can be written as follows:

$$\Lambda(\boldsymbol{\omega}_k) = \text{diag}\{|\omega_{k,1}|^{-1}, |\omega_{k,2}|^{-1}, ..., |\omega_{k,L}|^{-1}\} \quad (15)$$

which was proposed in [21].

3) Spatial Smoothing: In (3) the prior over the class labels can be written as follows which was first proposed in [22] and [23]:

$$p(\mathbf{z}_{1:N}|\beta) = \frac{\prod_{k=1}^{K}\exp\{\beta\sum_{n=1}^{N} z_{n,k}\left(1 + \frac{1}{2}\sum_{m\in\tilde{n}} z_{m,k}\right)\}}{\mathcal{Z}(\beta)} \quad (16)$$

where $\mathcal{Z}(\beta)$ is the normalization term, $\tilde{n}$ is the set of pixels around the n^{th} pixel, and β is the smoothing parameter. Smoothing is performed by maximizing the posterior distribution of pixel labels by using iterated conditional mode (ICM) algorithm [24]. Conditional of $\mathbf{z}_n$ can be maximized as follows:

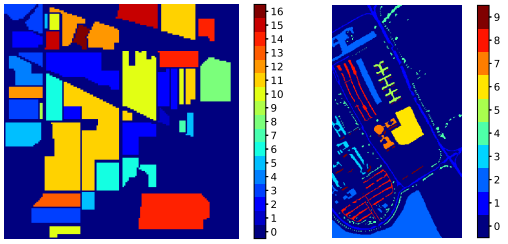
$$p(\mathbf{z}_n|\mathbf{z}_{\bar{n}}, \mathbf{s}_B, \hat{\boldsymbol{\omega}}_{1:K}, \beta) \propto p(\mathbf{s}_n|\mathbf{z}_n, \hat{\boldsymbol{\omega}}_{1:K})p(\mathbf{z}_n|\mathbf{z}_{\tilde{n}}, \beta) = \prod_{k=1}^{K}\left[\frac{e^{\hat{\boldsymbol{\omega}}_k^T\mathbf{s}_n}}{\sum_{j=1}^{K} e^{\hat{\boldsymbol{\omega}}_j^T\mathbf{s}_n}}\frac{e^{\beta v_{n,k}}}{\sum_{j=1}^{K} e^{\beta v_{n,j}}}\right]^{z_{n,k}} \quad (17)$$

where $\bar{n} = \{1, 2, ..., N\} \setminus \{n\}$, $v_{n,k} = 1 + \sum_{m\in\tilde{n}} z_{m,k}$, B is the set of test pixel indices, and $\hat{\boldsymbol{\omega}}_k$ is the estimated regression parameter vector of the k^{th} class.

III. Experimental Results

In order to evaluate the classification performance of the proposed method we performed tests with two hyperspectral images, namely Indian Pines and Pavia University.

Indian Pines image consists of 145×145 pixels, 16 classes, and 220 spectral bands. In experiments, we removed 20 noisy bands and used the remaining 200 bands. Pavia University image consists of 610×340 pixels, 9 classes and 103 spectral bands. In Fig. 1, the ground truth images of Indian Pines and Pavia University are given alongside color-bars showing the class numbers.



(a) (b)

Fig. 1. Ground truth images of (a) Indian Pines and (b) Pavia University datasets.

We have run the tests 20 times and report the average accuracy along with standard deviation. For each run, for the Indian Pines image, training set is constructed by randomly selecting 50 pixels from each class, and the remaining pixels are used as the test samples. If half of the number of pixels in a class is smaller than 50, half of the pixels are used for training and the other half is used for tests. The same procedure is applied to the Pavia University image except that we select 100 pixels from each class for training. The number of training and test pixels for Indian Pines and Pavia University are given in Table I and II, respectively.

TABLE I
INDIAN PINES CLASSES WITH NUMBER OF TRAINING AND TEST SAMPLES

Class	Name	Training	Test
1	Alfalfa	23	23
2	Corn-notill	50	1378
3	Corn-mintill	50	780
4	Corn	50	187
5	Grass-pasture	50	433
6	Grass-trees	50	680
7	Grass-pasture-mowed	14	14
8	Hay-windrowed	50	428
9	Oats	10	10
10	Soybean-notill	50	922
11	Soybean-mintill	50	2405
12	Soybean-clean	50	543
13	Wheat	50	155
14	Woods	50	1215
15	Buildings-Grass-Trees-Drives	50	336
16	Stone-Steel-Towers	46	47
Total		693	9556

TABLE II
PAVIA UNIVERSITY CLASSES WITH NUMBER OF TRAINING AND TEST SAMPLES

Class	Name	Training	Test
1	Asphalt	100	6531
2	Meadows	100	18549
3	Gravel	100	1999
4	Trees	100	2964
5	Painted metal sheets	100	1245
6	Bare Soil	100	4929
7	Bitumen	100	1230
8	Self-Blocking Bricks	100	3582
9	Shadows	100	847
Total		900	41876

As a preprocessing step, from the training data we subtract the mean and divide all the samples by the standard deviation. Using the mean and the standard deviation of the training set, we apply the same transformation to the test set as well.

For the Indian Pines image, we use a network with three 1-D convolution layers with number of filters 10, 10, and 5, and with kernel sizes 5, 3, and 5 in consecutive order. Each layer consists of a convolution operator, batch normalization, ReLU, and a max pooling operator with stride 2 and kernel size 2. The resultant feature length is 110. Number of iterations for training ASMLR is 50, and similarly, number of epochs for training CNN and CNN+ASMLR is also 50.

For the Pavia University image, we use 10, 10, and 15 convolutional filters consecutively. Since the spectral vector length is 103, we use a zero padding operation in the first layer before the max pooling layer, in order to make it divisible by 2. Other parameters are the same, and the resultant feature length is 150. Number of iterations used for training ASMLR is 100, and similarly, number of epochs to train CNN and CNN+ASMLR algoritms is also 100.

The number of filters, kernel sizes, and layers were decided by experiments for both networks. Pooling layers limit the depth of the network since the spectral dimension becomes smaller in each layer, therefore 3 layers were considered as optimum.

For comparison, we also use a support vector machine (SVM) which is combined with the same spatial smoothing algorithm. Although support vector machine is not a probabilistic classifier, we use Platt scaling [25] to get class probabilities, therefore it can be combined with the spatial smoothing algorithm.

The test results of Indian Pines and Pavia University images are given in Table III and Table IV, respectively.

TABLE III
INDIAN PINES TEST RESULTS

Evaluation Metrics	**Methods**			
	ASMLR	***CNN***	***CNN+ASMLR***	***SVM***
Accuracy	0.84	0.83	0.89	0.78
Standard deviation	0.03	0.06	0.03	0.03

TABLE IV
PAVIA UNIVERSITY TEST RESULTS

Evaluation Metrics	**Methods**			
	ASMLR	***CNN***	***CNN+ASMLR***	***SVM***
Accuracy	0.92	0.87	0.93	0.76
Standard deviation	0.02	0.11	0.05	0.02

From Table III and Table IV it can be seen that the proposed method gives better results than the other methods. The difference between the results obtained by CNN and CNN+ASMLR shows that the proposed procedure significantly improves the classification performance of CNNs.

Classification map examples of the Indian Pines image are given in Fig. 2.

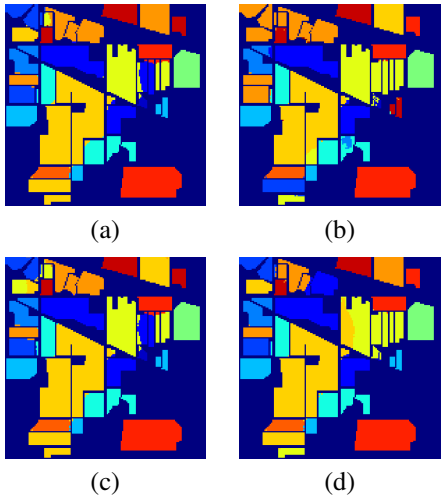


Fig. 2. Classification maps of Indian Pines image obtained by (a) ASMLR, (b) CNN, (c) CNN+ASMLR, and (d) SVM.

Classification map examples of the Pavia University image are given in Fig. 3.

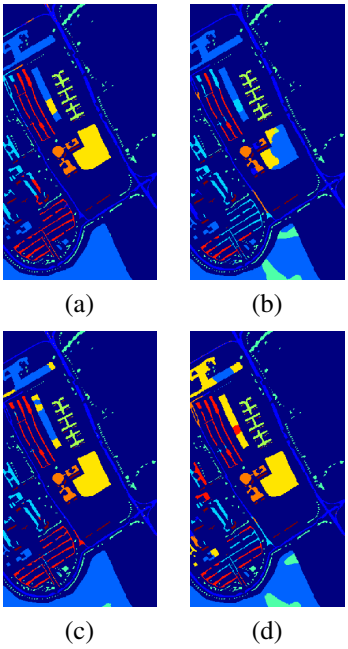


Fig. 3. Classification maps of Pavia University image obtained by (a) ASMLR, (b) CNN, (c) CNN+ASMLR, and (d) SVM.

IV. Conclusion

Experimental results have shown that the proposed training method is advantageous in terms of classification accuracy. This method combines the feature extraction capabilities of convolutional layers and classification performance of ASMLR. As can be seen from the reported test results, the combination of convolutional representation learning, sparsity based feature selection, and spatial smoothing improve the classification accuracy even when trained on a relatively small training set. The proposed procedure with ASMLR training can be applied to CNNs with different architectures and it allows to train CNNs with a smaller number of training iterations.

Future work will include a more detailed comparison of different network structures, performance evaluation with different training strategies, and modifications for minibatch training.

References

[1] G. Hughes, "On the mean accuracy of statistical pattern recognizers," *IEEE Transactions on Information Theory*, vol. 14, no. 1, pp. 55–63, January 1968.

[2] Y. Chen, H. Jiang, C. Li, X. Jia, and P. Ghamisi, "Deep feature extraction and classification of hyperspectral images based on convolutional neural networks," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 10, pp. 6232–6251, Oct 2016.

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

[4] P. Jia, M. Zhang, W. Yu, F. Shen, and Y. Shen, "Convolutional neural network based classification for hyperspectral data," in *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, July 2016, pp. 5075–5078.

[5] T. Alipourfard, H. Arefi, and S. Mahmoudi, "A novel deep learning framework by combination of subspace-based feature extraction and convolutional neural networks for hyperspectral images classification," in *2018 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 2018, pp. 4780–4783.

[6] J. Yang, Y. Zhao, J. C. Chan, and C. Yi, "Hyperspectral image classification using two-channel deep convolutional neural network," in *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, July 2016, pp. 5079–5082.

[7] H. Lee and H. Kwon, "Contextual deep cnn based hyperspectral classification," in *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, July 2016, pp. 3322–3325.

[8] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 1–9.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 770–778.

[10] F. I. Alam, J. Zhou, A. W. Liew, and X. Jia, "Crf learning with cnn features for hyperspectral image segmentation," in *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, July 2016, pp. 6890–6893.

[11] P. Ghamisi, Y. Chen, and X. X. Zhu, "A self-improving convolution neural network for the classification of hyperspectral data," *IEEE Geoscience and Remote Sensing Letters*, vol. 13, no. 10, pp. 1537–1541, Oct 2016.

[12] A. J. Simpson, "Dither is better than dropout for regularising deep neural networks," *arXiv preprint arXiv:1508.04826*, 2015.

[13] 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," *IEEE Transactions on Image Processing*, vol. 27, no. 5, pp. 2354–2367, May 2018.

[14] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 11, pp. 1222–1239, Nov 2001.

[15] K. Kayabol, "Approximate sparse multinomial logistic regression for classification," *IEEE Trans. Pattern Anal. Machine Intell.*, 2019, accepted for publication.

[16] S. Kutluk, K. Kayabol, and A. Akan, "A discriminative model for contextual classification of hyperspectral images," in *2018 26th Signal Processing and Communications Applications Conference (SIU)*, May 2018, pp. 1–4.

[17] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[18] X. X. Zhu, D. Tuia, L. Mou, G. Xia, L. Zhang, F. Xu, and F. Fraundorfer, "Deep learning in remote sensing: A comprehensive review and list of resources," *IEEE Geoscience and Remote Sensing Magazine*, vol. 5, no. 4, pp. 8–36, Dec 2017.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[20] D. Böhning, "Multinomial logistic regression algorithm," *Annals of the institute of Statistical Mathematics*, vol. 44, no. 1, pp. 197–200, 1992.

[21] B. Krishnapuram, L. Carin, M. A. Figueiredo, and A. J. Hartemink, "Sparse multinomial logistic regression: Fast algorithms and generalization bounds," *IEEE transactions on pattern analysis and machine intelligence*, vol. 27, no. 6, pp. 957–968, 2005.

[22] K. Kayabol and J. Zerubia, "Unsupervised amplitude and texture classification of sar images with multinomial latent model," *IEEE Transactions on Image Processing*, vol. 22, no. 2, pp. 561–572, Feb 2013.

[23] K. Kayabol and S. Kutluk, "Bayesian classification of hyperspectral images using spatially-varying gaussian mixture model," *Digital Signal Processing*, vol. 59, pp. 106 – 114, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1051200416301269

[24] J. Besag, "On the statistical analysis of dirty pictures," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 48, no. 3, pp. 259–302, 1986.

[25] J. Platt *et al.*, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," *Advances in large margin classifiers*, vol. 10, no. 3, pp. 61–74, 1999.