Multispectral Image Analysis using Convolution Neural Networks

Arun D. Kulkarni
University of Texas at Tyler, akulkarni@uttyler.edu

Multispectral Image Analysis using Convolution Neural Networks

Arun D. Kulkarni
Computer Science Department
The University of Texas at Tyler
Tyler, TX 75703, USA

***Abstract*—Machine learning (ML) techniques are used often to classify pixels in multispectral images. Recently, there is growing interest in using Convolution Neural Networks (CNNs) for classifying multispectral images. CNNs are preferred because of high performance, advances in hardware such as graphical processing units (GPUs), and availability of several CNN architectures. In CNN, units in the first hidden layer view only a small image window and learn low level features. Deeper layers learn more expressive features by combining low level features. In this paper, we propose a novel approach to classify pixels in a multispectral image using deep convolution neural networks (DCNNs). In our approach, each feature vector is mapped to an image. We used the proposed framework to classify two Landsat scenes that are obtained from New Orleans and Juneau, Alaska areas. The suggested approach is compared with the commonly used classifiers such as the Decision Tree (DT), Support Vector Machine (SVM), and Random Forest (RF). The proposed approach has shown the state-of-the-art results.**

Keywords—Convolution neural networks; machine learning; multispectral images; remote sensing

I. Introduction

Recently there has been a great interest in the research community to adapt CNNs to analyze multispectral images. Machine learning algorithms such as the decision tree (DT), ensemble of decision trees, support vector machine (SVM), Naïve Bayes classifier and fuzzy inference system are used to classify pixels in multispectral images. CNN-based methods have attracted a great deal of attention due to their ability to dig latent representations and features from images. Borisov, et al. [1] in their survey article provided an overview of deep learning methods for tabular data. They point out that CNN models have repeatedly shown excellent performance and have been widely adapted. However, adaptation of CNN models to tabular data remains highly challenging. DCNNs have shown high accuracy in many image classification applications. They are flexible and allow iterative learning. With advances in hardware and availability of graphical processing units (GPUs) deep learning (DL) models can be used in real-life applications. The main drawback of CNN models is that they commonly use gradient decent backpropagation algorithm with Sigmoid activation functions that leads to saturation resulting in slow gradient convergence, which is known as a vanishing gradient problem. To avoid the vanishing gradient problem, CNNs use entropy loss function with rectified linear units (ReLU) in the output layer. Overfitting usually occurs when the dataset is small. To overcome this problem, various regularization techniques such as dropout and bagging are used. CNN models can be trained with large datasets and can classify images with high accuracy. For ML algorithms, data are presented in the table form whereas for CNN input data are presented in the form of images. CNN models generate a feature vector from input images via convolution and pooling layers. The feature vector represents integrated information from various shapes that are present in the input image.

To use available CNN models and take advantage of these models such as high accuracy, we propose a novel approach to convert data in the table form to images. In ML algorithms input data are in table form, where each row in the table represents a feature vector for an entity and columns describe properties. In many fields such as genomics, transcriptomic, spoken words, financial and banking data are in non-image form. A few researchers have proposed techniques to map numeric data in a table form to images [2 - 4].

We propose a method to map a feature vector into an image. The shapes in the mapped image represent features and ratios between the features. The approach was motivated by two factors. The first motivational factor is that band ratios in Landsat images are used for determining soil moisture coefficients and vegetation indices [5], which indicates that feature ratios contain additional information in the sample than features alone. The second motivational factor is that CNN models provide multiple levels of abstraction and generate a feature vector that combine low-level and high-level information of shapes in the image. To validate our framework, we analyzed two Landsat-8 scenes. Landsat-8 OLI provides images with 30-meter resolution in seven spectral bands. Band 1 reflects deeper blue-violet hues and is used in mapping coastal regions. Bands 2, 3, and 4 are visible blue, green, and red are used in land use mapping. Band 5 measures the near infrared (NIR). Bands 6 and 7 cover different slices of the shortwave infrared (SWIR). They are particularly useful for telling wet earth from dry earth, and for geology [6]

We implemented the algorithm to map table data into images using a MATLAB script. In addition, we implemented Alex Net using MATLAB deep learning toolbox. We analyzed two scenes, one from New Orleans, and another from Juneau, Alaska. To classify pixels in the image each pixel is represented by a vector consisting of reflectance values in multiple spectral bands. We extracted training set data by displaying scenes on the monitor and selecting small homogeneous areas that represent distinct categories on the ground. We selected four training areas that represented four categories for the Alaska scene, and three training areas that represented three categories for the New Orleans scene. We selected reflectance values from spectral bands 2, 3, 4, 5, and 7, as these bands showed the maximum variance. The organization of the paper is as follows. Section II describes the related work. Section III provides the framework for the proposed approach. Section IV deals with the experiment and results, and the whole study is concluded in Section V.

II. Related Work

In remote sensing data are obtained as multispectral images. Many ML algorithms are being used to classify pixels in a multispectral image. The conventional ML techniques for classification require the sample in the form of a feature vector. In classifying Landsat-8 scene, each pixel is represented by a feature vector obtained by reflectance values in different spectral bands. A few small homogeneous areas representing distinct categories on the ground are identified and the feature vectors associated with pixels in those training areas are used to generate training set data. Conventional ML techniques include the maximum likelihood classifier (MLC), decision tree (DT), support vector machine (SVM), Random Forest (RF), multi-layer perceptron model, and fuzzy inference system. The maximum likelihood classifier assumes normal distribution of reflectance values and is commonly used in remote sensing applications. Huang et al. [7] have used the SVM to classify pixels in multispectral images and their model obtained higher accuracy. The SVM is appealing for Landsat data analysis because it classifies small data sets with high accuracy [8]. Moumtrakis et al. [9] have provided a review of usage of SVM in remote sensing. Another commonly used algorithm to classify pixels in multispectral images is a decision tree (DT). The main problem with the DT is overfitting. DT shows high accuracy with the training data, however; it may not perform well in classifying unknown data samples. Lowe and Kulkarni [10] used the random forest algorithm for classification of pixels in Landsat data.

CNN models represent one of the best learning algorithms for understanding image contents and have shown exemplary performance in computer vision tasks. CNN models use multiple layers of nonlinear information processing units. Machine-learning community's interest in CNN grew after Image-Net competition in 2012, where Alex Net achieved record breaking results in classifying images from the dataset containing more than 1.2 million images from one thousand classes. Alex Net proposed by Krizevsky et al. [11] was based on principles used in LeNet. DCNNs have brought about breakthroughs in processing images, videos, speech, and audio [12]. In general CNN models consists of convolution and pooling layers that are grouped followed by one or more fully connected layers. They are feed-forward networks. In convolution layers, inputs are convolved with a weighted kernel and the output is sent via a nonlinear activation function to the next layer. The purpose of the pooling layer is to reduce spatial resolution. Rawat and Wang [13] provide a comprehensive survey of CNNs. Zhang et al. [14] provide taxonomy of CNN models. CNNs can learn internal representations from raw pixels and are hierarchical learning models that can extract features [15, 16]. Liu et al. [17] provide a survey of deep neural network architectures and their applications. Khan et al. [18] in their review article classified DCNN architectures into seven categories. Deep learning allows computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction. Recent developments in CNN models were possible because of the availability of faster graphical processing units (GPUs) and availability of large data sets. Kulkarni in [19] used the Alex Net to classify two image datasets. The first dataset contained four hundred animal images of two types of animals and obtained 99.1 percent accuracy. The second dataset contained four thousand images of five types of flowers and obtained 86.64 percent accuracy.

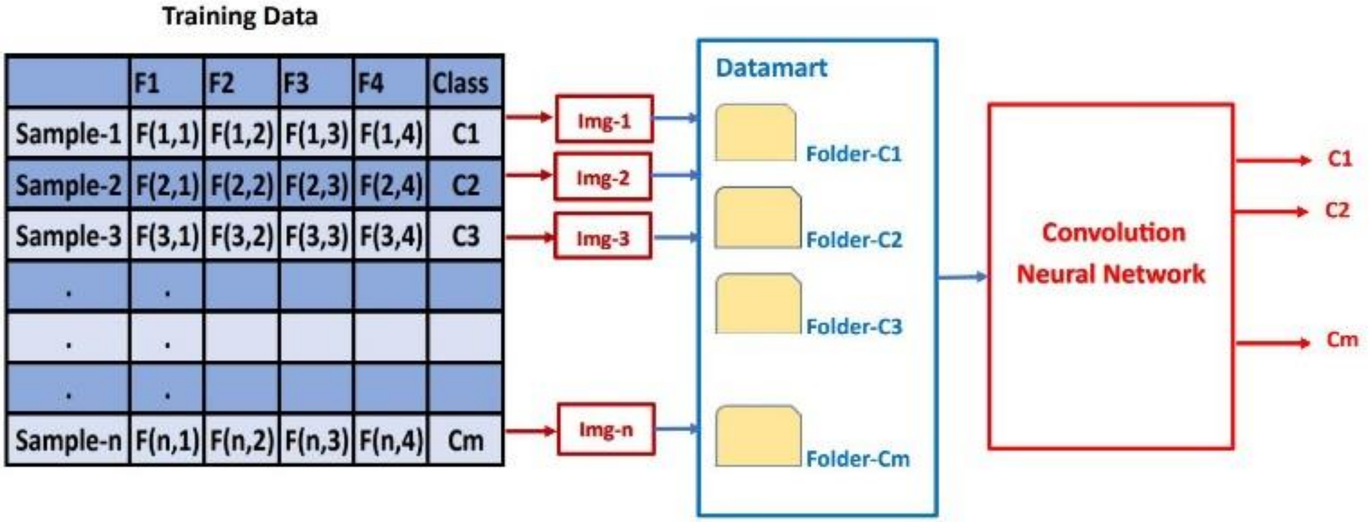
Remote sensing images are often obtained in multiple spectral bands. Traditional ML are used to classify pixels in multispectral images, where each pixel is represented by a feature vector consisting of reflectance values from different spectral bands. There is a great deal of research directed towards CNN architectures for RGB images, while a relative dearth of research directed towards CNN architectures for multispectral and hyperspectral images. Many DCNN models have been deployed to analyze remote sensing data. Castelluccio et al. [20] explored the use of CNN models for semantic classification of remote sensing scenes. They resort to pre-trained CNNs that are only fine-tuned on the target data to avoid overfitting problems and reduce design time. Liu [21] used R-CNN for multispectral pedestrian detection task and then modeled it into a convolution network (ConvNet) for the fusion problem. Xu et al. [22] proposed a CNN framework to extract spectral-spatial features from hyperspectral imagery (HIS) and light detection and ranging (LiDAR) data, and to combine HIS and LiDAR data. Chen et al. [23] used Faster R-CNN for airport detection from Landsat images. Their experimental results show that for the same training samples their CCN based approach outperforms traditional SVM and state-of-the-art CNN based methods. Senecal et al. [24] have created a small CNN architecture capable of being trained from scratch to classify 10 band multispectral images. Osorio et al. [25] used a deep learning approach for weed detection. They used a method YOLOV3 (you only look V3), taking advantage of robust architecture for object detection. Garcia et al. [26] studied the use of different CNN architectures for cloud masking in multispectral images. Yuan et al. [27] proposed a novel DCNN architecture that outperforms the state-of-the-art DCNN-based water body detection methods. Wu et al. [28] proposed a deep-learning-based new framework for multimodal remote sensing data classification.

Tabular data are the most used data. DCNN models have shown excellent performance and have therefore been widely adapted. However, their adaptation to tabular data remains highly challenging. Borisov, et al. [1] provide an overview of deep learning methods for tabular data. They categorize these methods into three groups a) data transformations, b) specialized architectures, c) regularization models. In this work we consider the first category data transformations. DCNNs offer multiple advantages over traditional ML techniques. First, they are flexible and allow iterative learning. Second, tabular data generation is possible using deep networks and can help mitigate class imbalance problems. Third, neural networks can be deployed for multimodal learning problems where tabular data can be one of many input modalities [1]. To our knowledge there are very few methods that have been reported for analyzing tabular data using CNN models. Sharma et al. [2] developed a method called DeepInsight to transform non-image data to images for CNNs. Their method constructs an image by placing similar features together and dis-similar ones further apart enabling the collective use of neighboring elements. This collective approach of element arrangements can be useful in understanding relationships between a set of features. They employed four distinct kinds of datasets to evaluate their algorithm. They compared the obtained results to state-of-the art classifiers such as the decision tree, Ada-boost, and random forest. Their model had shown better classification accuracy for all datasets. Buturovic and Mitkovic [29] proposed algorithm called TAC (table to convolution) to embed a feature vector into image. They used the base-image, and the feature vector is used as a kernel to obtain the convolved image. Zhu et al. [3] have suggested a method called Image Generator for Tabular Data (IGTD) to transform tabular data into images by assigning features to pixel positions so that similar features are close to each other. The algorithm assigns each feature to a pixel in the image. An image is generated for each data sample, in which the pixel intensity reflects the value of the corresponding feature in the sample. The algorithm searches for the optimized assignment of features to pixels by minimizing the difference between the ranking of the pairwise distances between features and the raking pairwise distances between assigned pixels. To investigate the utility of the IGTD, they applied the algorithm to two datasets CCL gene expression and drug molecular descriptors. They transformed these tabular datasets into images and classified them using CNN. Their results show that the CNNs trained on IGTD images provide the highest average prediction performance in cross-validation on both datasets. Kulkarni [30] has proposed a method to map tabular data into images. Each feature vector is mapped to an image. The number of mapped regions in the image is equal to the number of features and the gray values of the regions represent feature values. The algorithm was used to classify a dataset representing URLs for phishing detection. Sun et al. [4] have proposed a method to convert tabular data into images called SuperTML. The algorithm borrows the concept of the Super Characters method to address tabular data machine learning tasks. For each input tabular features are first projected onto a two-dimensional embedding and fed into fine-tuned two-dimensional CNN models for classification. They validated the algorithm by using four datasets. Their experimental results show that SuperTML method has achieved state-of-the-art results on both large and small tabular datasets.

III. Proposed Framework

In machine learning approaches such as decision tree, support vector machine, models are trained using feature vectors from the training set data. In our method, we convert feature vectors into images, which are saved in DataMart. Images are saved in the folders that are labeled with class names. The CNN model is trained with images in Datamart. The framework for the CNN training is shown in Fig. 1. The crucial step in the proposed approach is to convert a feature vector into a 2-D image matrix. In the proposed approach the output image contains n^2 rectangles, where n represents the number of features in the feature vector. For example, if there are five features in the feature vector, the corresponding output image contains twenty-five rectangular shapes. The areas diagonal squares represent the features, and the areas of the off-diagonal rectangles represent ratios of the features as shown in Eq. (1), where A_{ij} represents the area of the shape, i and j represent. the row and the column numbers of the shape and f_i represents a feature i.

$$A_{ij} = \begin{cases} \left(\dfrac{f_i}{f_j}\right) & if\ i \neq j \\ \left(f_i^2\right) & if\ \ i = j \end{cases} \tag{1}$$



Training Data

	F1	F2	F3	F4	Class
Sample-1	F(1,1)	F(1,2)	F(1,3)	F(1,4)	C1
Sample-2	F(2,1)	F(2,2)	F(2,3)	F(2,4)	C2
Sample-3	F(3,1)	F(3,2)	F(3,3)	F(3,4)	C3
.	.				
.	.				
.	.				
Sample-n	F(n,1)	F(n,2)	F(n,3)	F(n,4)	Cm

Fig. 1. Framework for CNN learning.

It can be seen from Fig. 2 that the widths and heights of diagonal rectangles represent feature values. The widths of the off-diagonal rectangles represent feature values, and heights of the off-diagonal rectangular shapes represent inverse of feature values.

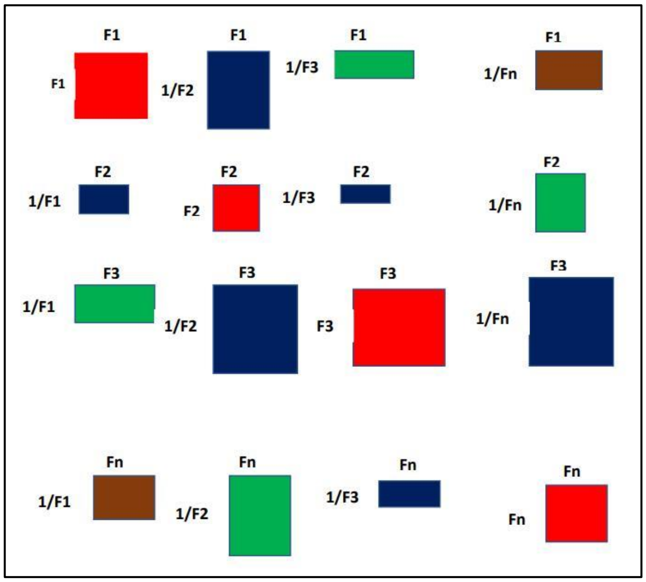


Fig. 2. Mapped image with n features.

Layers of a DCNN include the input layer, convolution layer, batch normalization layer, ReLU layer, max pooling layer, fully connected layer, SoftMax layer, and classification layer. We can specify the input image size at the input layer. Convolution layers serve as feature extractors. Inputs are convolved with learned weights to compute feature maps and results are sent through a nonlinear activation function. The output of the k^{th} feature map is given in Eq. (2).

$$Y_k = f\left(W_k * x\right) \quad (2)$$

where, x denotes the input image, W_k is the convolution filter. The '*' sign refers to the 2-D convolution operator [13] The batch normalization layers normalize the activations and gradients propagating through the network, which makes the training an easier optimization problem. The batch normalization layers are followed by ReLU layers. The purpose of the pooling layer is to reduce the spatial resolution by downsizing and extract invariant features. The max pooling layer is used to downsize the network and extract features. The fully connected, SoftMax and classifier layers map the feature vector to class labels. The output of the SoftMax layer consists of positive numbers that sum up one that are used as class probabilities. We used Alex Net to classify feature vectors. Layers of Alex New are shown in Fig. 3. The model consists of eight layers: five convolution layers and three fully connected layers.

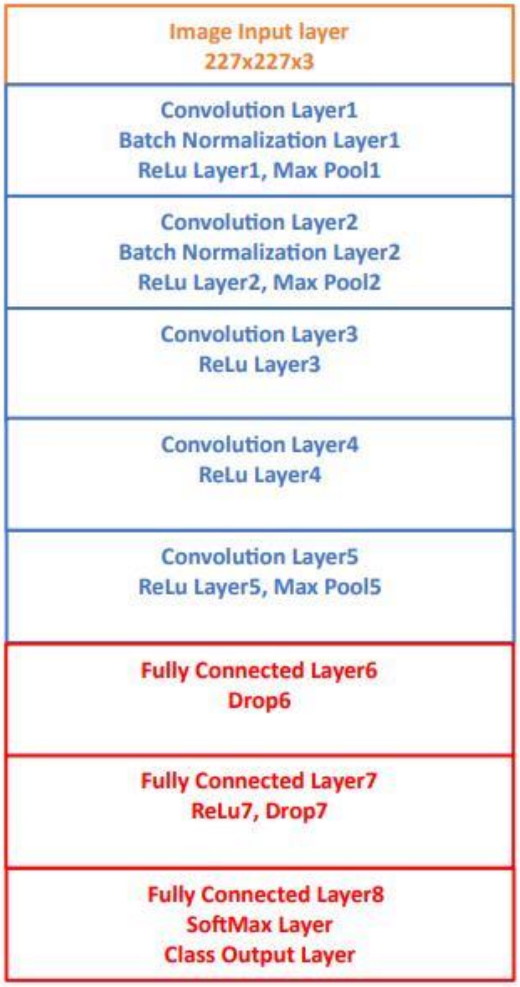


Fig. 3. Layers of alex net.

IV. Experiment and Results

We developed software to map tabular data into images using MATLAB script. The output images were stored in their respective class folders. We also implemented Alex Net using the MATLAB toolbox. The DCNN model was trained with images that were generated from feature vectors in tabular training set data. We classified pixels from two sub-scenes using the trained DCNN model. We considered Landsat-8 scenes that were obtained by Operational Land Imager (OLI).

A. *Example-1 New Orleans Scene*

The scene was obtained by Landsat-8 OLI on February 26, 2016. The path and row numbers for the scenes are 22 and 39, respectively. To generate the training set data, we considered the scene of the size 1000 by 1000 pixels. Three small homogeneous areas were selected as training sets that represent three classes water, land, and vegetation. The training set data contains 600samples, 200 from each class. We selected band-2, band-3, band-4, band-5, and band-7. The spectral signatures obtained from mean vectors of the classes are shown in Fig. 4. During training, feature vectors are mapped to images. Each image represents a feature vector representing a pixel in the multispectral image. The mapped images were stored in their respective class folders. These images were used to train and validate Alex Net. We used 70 percent randomly selected images for training and 30 percent for validation. With Alex Net we obtained the overall accuracy of 98.33 percent. The learning progress curve for Alex Net is shown in Fig. 5. The confusion matrix is shown in Fig. 6 and the ROC curves are shown in Fig. 7. We used the trained network models to classify sub-scene of the size 256 by 256 pixels. Fig. 8 shows the classified scenes for the New Orleans area.

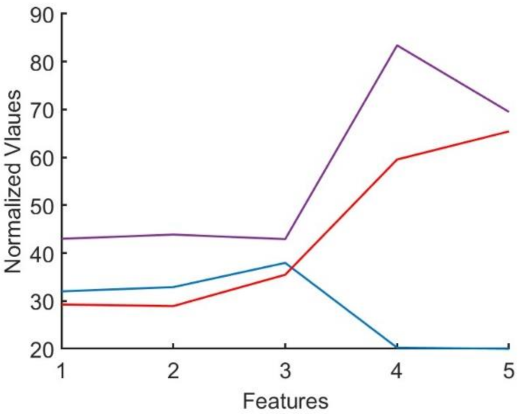


Fig. 4. Spectral signatures for New Orleans scene.

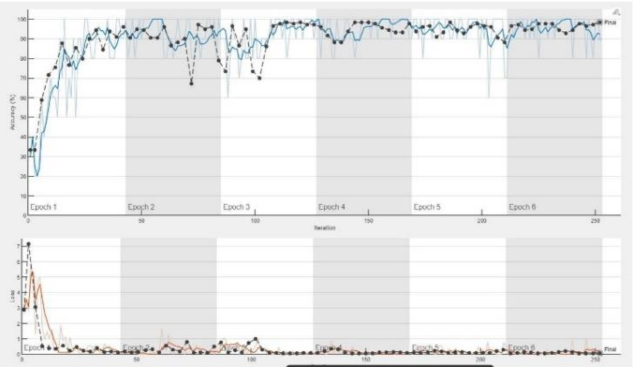


Fig. 5. Training progress curve for New Orleans scene.

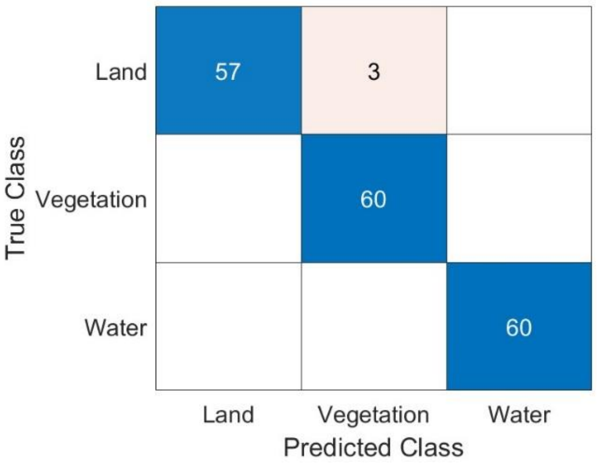


Fig. 6. Confusion matrix New Orleans scene.

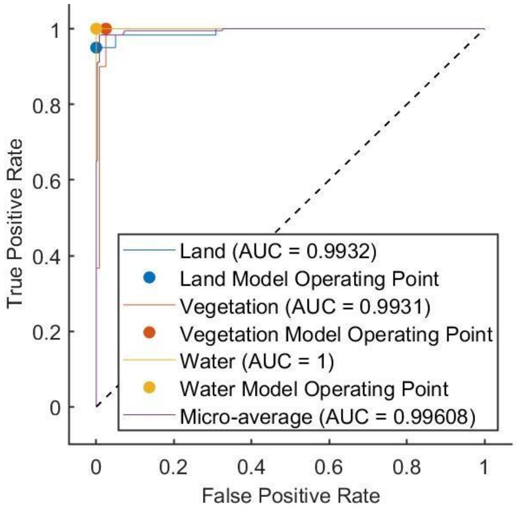


Fig. 7. ROC curve New Orleans scene.

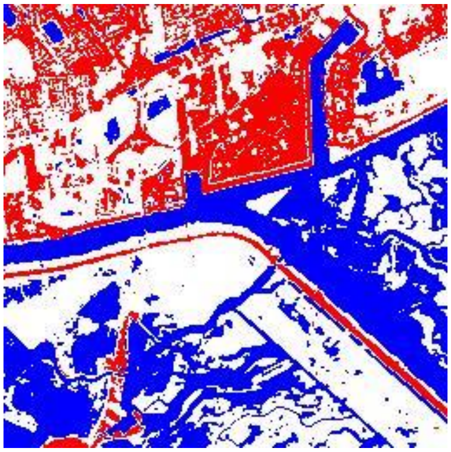


Fig. 8. Classified New Orleans scene.

B. *Example-2 Juneau Alaska Scene*

The scene was obtained by Landsat-8 OLI on June 13, 2016. The path and row numbers for the scene are 58 and 19, respectively. To generate the training set data, we considered the scene of the size 1000 by 1000 pixels. Four small homogeneous areas were selected as training sets that represent four classes' water, vegetation, ice-land, and glaciers.

The training set data contains 400 samples, 100 from each class. We selected band-2, band-3, band-4, band-5, and band-7 as features. The spectral signatures obtained from mean vectors of the classes are shown in Fig. 9. During training, feature vectors mapped into images. Each image represents a feature vector representing a pixel in the multispectral image. The mapped images were stored in their respective class folders. These images were used to train Alex Net. We used 70 percent randomly selected images for training and 30 percent for validation. We obtained the overall accuracy of 98.33 percent. The learning progress curve for Alex Net is shown in Fig. 10. The confusion matrix is shown in Fig. 11 and ROC curves are shown in Fig. 12. We used a trained network model to classify sub-scene of the size 256 by 256 pixels. The classified scene is shown in Fig. 13.

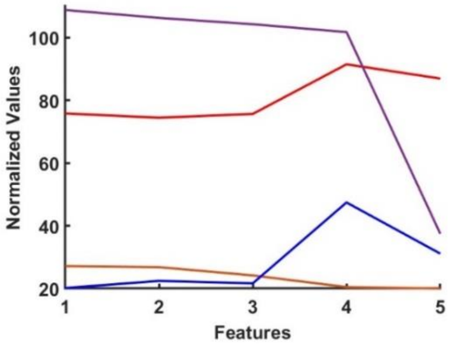


Fig. 9. Spectral signatures for Alaska scene.

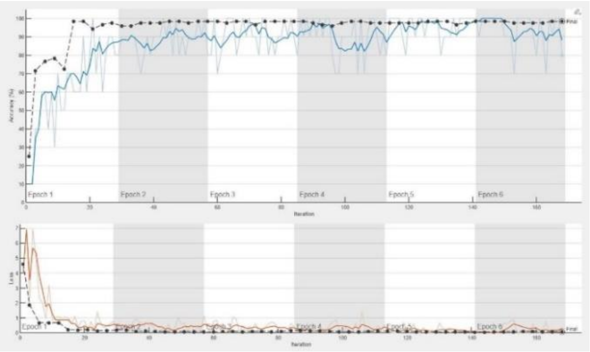


Fig. 10. Training progress curve Alaska scene.

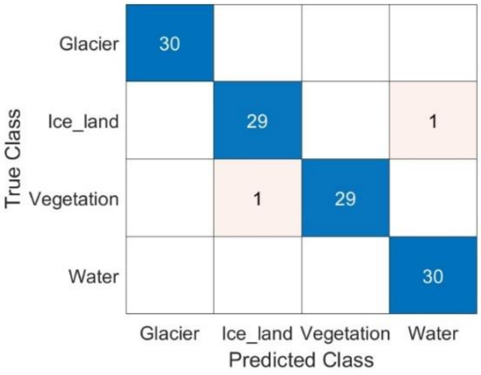


Fig. 11. Confusion matrix Alaska scene.

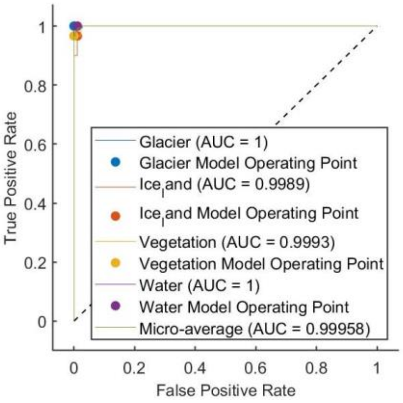


Fig. 12. ROC curve Alaska scene.

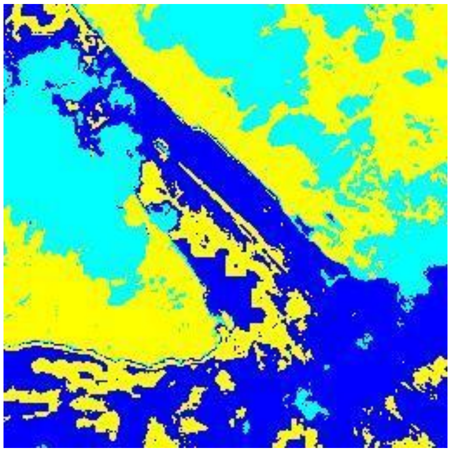


Fig. 13. Classified Alaska scene.

C. Comparison of Classifier Results

We also classified both datasets using DT, SVM, and RF algorithms. The results are shown in Table I. It can be seen from the results that the proposed algorithm provides state-of-the-art results.

TABLE I. OVERALL ACCURACY

Dataset	***Decision Tree***	***Support Vector Machine***	***Random Forest***	***Alex Net***
New Orleans Scene	98.33 %	97.50 %	97.50 %	98.33
Alaska Scene	96.11 %	97.22 %	98.89	98.33

V. CONCLUSIONS

In this paper, we proposed a new method to classify tabular data using CNNs. We used the proposed method to classify pixels in a multispectral image. We developed the algorithm and implemented it using MATLAB script. We analyzed two Landsat-8 scenes, one from the New Orleans area and another from the Alaska area. The training set data for these scenes were generated by selecting small homogeneous areas from the displayed scenes. Feature vectors were mapped to images that were used to train the DCNN model. The results show that the proposed approach yields the state-of-the-art results. The limitation of the proposed approach is that the number of features that can be processed. This is due to the limit on the number of rectangular regions that can be accommodated in a transformed image. The maximum size of the transformed image is finite which limits the number of shapes in the transformed image. The future work includes using other complex shapes to represent feature values and using DCNN models such as Resnet-50 and Google Net so that the overall accuracy can be improved. Also, we would like to extend the algorithm for remote sensing images with a greater number of spectral bands.

REFERENCES

[1] V. Borisov, T. Leemann, K. Seßler, J. Haug, M. Pawelczyk and G. Kasneci, "Deep Neural Networks and Tabular Data: A Survey," in IEEE Transactions on Neural Networks and Learning Systems, doi: 10.1109/TNNLS.2022.3229161.

[2] A. Sharma, E. Vans, D. Shigemizu, K. A. Boroevich, and T. Tsunoda, “Deep Insight: A methodology to transform a non-image data to an image for convolution neural network architecture.” Nature Sci Rep 9, 11399, 2019.

[3] Y. Zhu, T.Brettin, F. Xia, A. Partin, M. Shukla, H. Yoo, Y. A. Evrard, J. H. Doroshow, and R. L. Stevens, “Converting tabular data into images for deep learning with convolutional neural networks” Nature Scientific Reports, 2021 https://doi.org/10.1038/s41598-021-90923-y

[4] B. Sun et al., "SuperTML: Two-Dimensional Word Embedding for the Precognition on Structured Tabular Data," 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Long Beach, CA, USA, 2019, pp. 2973-2981, doi: 10.1109/CVPRW.2019.00360.

[5] D. N. Thi, N. T. T. Ha, Q. T. Dang, K. Koike, and N. M. Trong, “Effective band ratio of Landsat 8 Images based on VNIR-SWIR reflectance spectra of topsails for soil moisture mapping in a tropical region, Remote Sens. 2019, vol. 11, no. 6, 716; https://doi.org/10.3390/rs11060716

[6] USGS-Landsat-8 https://www.usgs.gov/landsat-missions/landsat-8

[7] C. Huang, L. Davis, and J. Townshend, “An assessment of support vector machines for land cover classification,” International Journal of Remote Sensing, vol. 23, no. 4, pp. 725-749, 2002.

[8] P. Mantero, G. Moser, and S. B. Serpico, “Partially supervised classification of remote sensing images through SVM-based probability density estimation,” IEEE Transactions on Geoscience and Remote Sensing, vol. 43, no. 3, pp. 559-570, 2005.

[9] G. Mountrakis, J. Im, and C. Ogole, “Support vector machines in remote sensing: A review,” ISPRS Journal of Photogrammetry and Remote Sensing, vol. 66, no. 3, pp. 247-259, 2011.

[10] B. Lowe and A. D. Kulkarni. “Multispectral image analysis using Random Forest,” International Journal on Soft Computing (IJSC), vol. 6, no.1, pp. 1-14, 2015.

[11] A. Krizhevsky, I. Sutskever, G. Hinton, “ImageNet classification with deep convolutional neural networks.” Adv Neural Inf Process Syst. 2012 https://doi.org/10.1061/(ASCE)GT.1943-5606.0001284

[12] Y. LeCun, Y. Bengin, and G. Hinton, “Deep learning,” Nature, vol. 521, pp. 436-444, 2015.

[13] W. Rawat, and Z. Wang, “Deep convolution neural networks for image classification: A comprehensive review,” Neural Computation, vol. 29, pp. 2352-2449, 2017

[14] S. Zhang, L. Yaq, A. Sun, Y. Tay, “Deep Learning based Recommender System: A Survey and New Perspectives,” ACM Computing Surveys, 2018, vol. 1, no. 1, pp. 1-35.

[15] Q. Abbas, M. Ibrahim, M. Jaffar (2019) A comprehensive review of recent advances in deep vision systems. Artif Intell Rev, 2019, vol. 52, pp 39–76. https://doi.org/10.1007/s10462-018-9633-3

[16] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, “Deep learning for visual understanding: A review,” Neurocomputing, 2016, vol. 187, pp. 27–48.

[17] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. F. Alsaadi, “A survey of deep neural network architectures and their applications,” Neurocomputing, 2017, vol. 234, pp.11-26.

[18] A. Khan, A. Sohail, U. Zahoora, A. S. Qureshi, “A Survey of the Recent Architectures of Deep Convolutional Neural Networks,” Artificial Intelligence Review, 2020, vol. 53, pp. 5455–5516. https://doi.org/10.1007/s10462-020-09825-6.

[19] A. D. Kulkarni, “Deep Convolution Neural Networks for Image Classification,” International Journal of Advanced Computer Science and Applications, Vol. 13, No. 6, pp 18-23, 2022.

[20] M. Castelluccio, G. Poggi, C. Sansone, and L. Verdoliva, “Land use classification in remote sensing images by convolutional neural networks,” arXiv:1508.00092v1 [cs.CV] 1 Aug 2015

[21] J. Liu, S. Zhang, S. Wang, D. N. Metaxas, “Multispectral Deep Neural Networks for Pedestrian Detection,” Computer Vision and Pattern Recognition, arXiv:1611.02644[cs.CV] https://doi.org/10.48550/arXiv.1611.02644

[22] X. Xu, W. Li, Q. Ran, Q. Du, L. Gao, B. Zhang, “Multisource remote sensing data classification based on convolutional neural network,” IEEE Transactions on Geoscience and Remote Sensing, vol. 56, no. 2, 2018, pp 937-949.

[23] F. Chen, R. Ren, T. V. Voorde, W. Xu, G. Zhou, and Y. Zhou, “Fast automatic airport detection in remote sensing images using convolutional neural networks,” Remote Sens. 2018, vol. 10, 443; doi:10.3390/rs10030443

[24] J. J. Senecal, J. W. Sheppard, and J. A. Shaw, “Efficient convolutional neural networks for multi-spectral image classification,” IJCNN 2019. International Joint Conference on Neural Networks. Budapest, Hungary. 14-19 July 2019, paper N-19045.

[25] K. Osorio O, A. Puerto, C. Pedraza, D. Jamaica, and L. Rodríguez, “A Deep Learning Approach for Weed Detection in Lettuce Crops Using Multispectral Images,” Agri Engineering vol 2, No. 3, pp. 471-488, 2020; https://doi.org/10.3390/agriengineering2030032

[26] G. M. Garca, L. G. Chova, G. C. Valls, “Convolution neural networks for multispectral image cloud masking,” arXiv:2012.05325v1 [cs.CV], 2020.

[27] K. Yuan, X. Zhuang, G. Schaefer, J. Feng, L. Guan, and H. Fang, “Deep-learning-based multispectral satellite image segmentation for water body detection, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 14, 2021, pp 7422-7434.

[28] X. Wu, D. Hong, and J. Chanussot, "Convolutional Neural Networks for Multimodal Remote Sensing Data Classification," in IEEE Transactions on Geoscience and Remote Sensing, vol. 60, pp. 1-10, 2022, Art no. 5517010, doi: 10.1109/TGRS.2021.3124913.

[29] L. Buturovic and D. Miljkovic, “A novel method for classification of tabular data using convolution neural networks” 2020. https://doi.org/10.1101/2020.05.02.074203

[30] A. D. Kulkarni “Convolution Neural Networks for Phishing Detection,” International Journal of Advanced Computer Science and Applications, vol. 14, no. 4, 2023, pp 15-19.