Terrain Classification via Deep Neural Networks

Lokesh Mishra

01.05.2019

Abstract : This document records the author's brief work on Terrain Classification. It starts with an introduction to the problem of terrain classification, provides a survey of important techniques and details the author's attempt at a solution (using a sequential deep neural network). Since this document shall serve as a personal reference (for future), an attempt is made towards completeness in explanations.

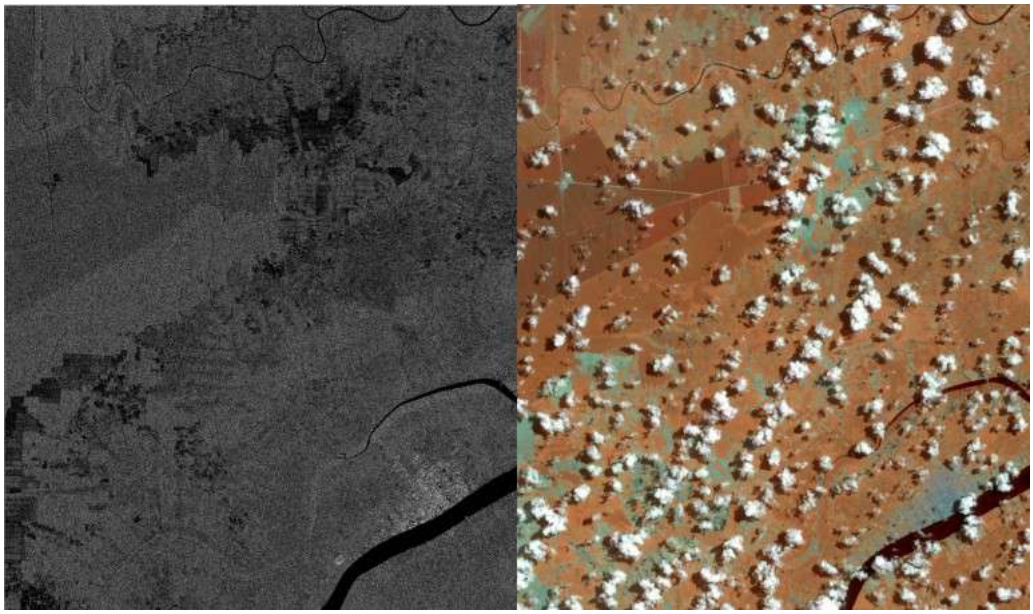

Figure 1: A comparison of the same area in Sumatra, Indonesia, collected using a radar sensor (Sentinel-1A) on the left and an optical sensor (Landsat-8) on the right. The cloud-penetrating capabilities of radar offer a great advantage in tropical areas that are frequently covered in clouds. Radar also reveals the textures of different land-cover types, making it easier to identify natural forest, farmlands, urban areas, roads and other land uses. (Left: Copernicus Sentinel data (2015), right U.S. Geological Survey.). From: https://bit.ly/2VhyEuZ

Contents

1 Introduction

For the purposes of this document, Terrain Classification may be understood as a subset of the well-known problem of Land Use and Land Cover (LULC) classification. A working definition for Land Cover states that it is the observed (bio-)physical cover of the earth's surface [4]. Furthermore, Land Use refers to the utilitarian purpose served by a land for socio-economic activities. Information generated from LULC classification serves a wide variety of purposes. To whet, consider these non-exhaustive examples: monitoring vegetation, natural resource management (oil, timber, mining, etc), disaster management (flooding, mud-slides, etc), urban expansion, wildlife habitat protection, environmental monitoring, earth radiation budgeting (eg: NASA CERES), humanitarian programs of agencies like UN, defence/military surveys et cetera.

Primarily, there are two methods for gathering LULC data: field survey and remote sensing. Field survey, as the name suggests, consists of data collection at the local level "in the field". Given the enormous geographic regions and lack of experts for analysis, this method is quite costly. Additional drawbacks, for this method, include large time consumption and lack of automation, among others [11]. On the other hand, remote sensing refers to the collection of data via satellites or aerial surveys. A major advantage of remote sensing is its capability to easily generate data in different resolutions (temporal, spectral and spatial). The quest for smarter ways to achieve LULC classification from data acquired via remote sensing has become an active area of research. In section 2, the various applications of deep learning towards this task will be presented. For the rest of this document, it is assumed that all LULC data is collected via remote sensing.

The number of classes recognized by LULC classifications can vary widely depending on the purposes, resources and research goals of the organization conducting said classification. For example, the Brazilian Coffee Scenes Dataset recognizes only two classes: coffee and non-coffee [10]. On the other hand, the Functional Map of the World (fMOW) challenge from the Intelligence Advanced Research Projects Agency (IARPA) recognized 63 classes [11]. Some of these classes are: airstrips, oil and gas facilities, mines, shipyards, ports, wind farms, office building, police station, crop field, solar farm, swimming pool, golf course, dam, hospital etc.

Terrain Classification, as presented here, will recognize 5 classes: Desert, Mountain, Forest, Water and Polar Caps [1]. A sample of these terrains can be seen in Figure 2. Before

[1] These 5 classes are only a small subset of the several classes recognized by LULC classification. Hence, it was suggested that Terrain Classification could be seen as a subset of the LULC classification problem.

describing the details of this problem in section 3, a general survey of various methods already applied for image recognition and LULC/Terrain classification is presented in the next section.

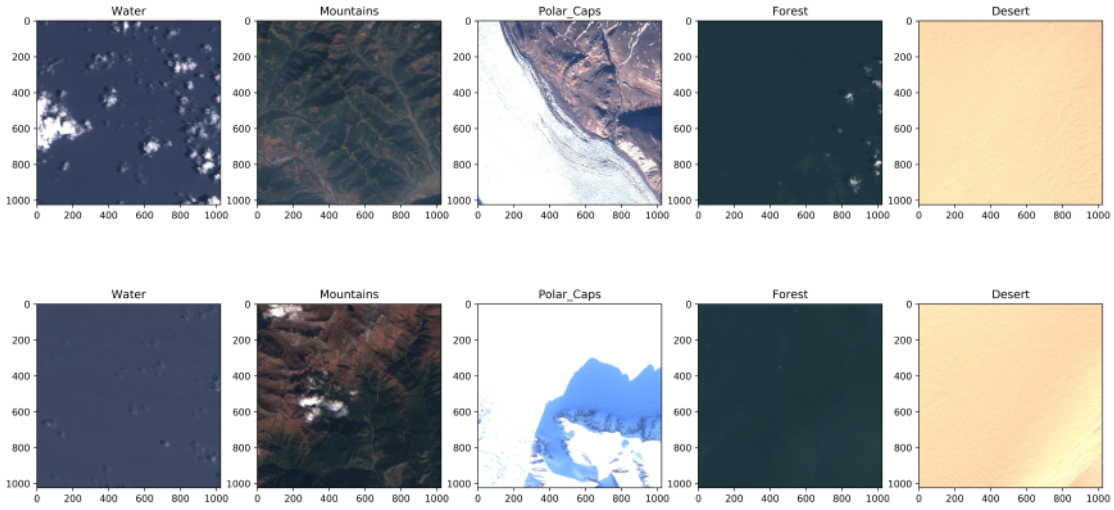

Figure 2: Sample images from the 5 classes considered in Terrain Classification

2 Survey of LULC Classification

LULC classification via Deep Learning usually employs three kinds of architectures[2]:

1. Deep Belief Network (DBN)
2. Autoencoder (AE)
3. Convolutional Neural Network (CNN)

A brief description of these architectures, as well as some of the reported Overall Accuracies (OA) is presented. The following is not a complete list, but it should give a general idea on the current performance of each architecture.

2.1 Deep Belief Network

DBN are based on artificial neural networks where the algorithm learns to classify using the hierarchical learning method [7]. The algorithm learns data layer-by-layer in an unsupervised greedy manner (also called pretreatment)[13]. Each layers is composed of a Restricted Boltzmann Machine (RBM). Finally, there is a softmax classifier in the last layer which fine-tunes the parameters in a supervised manner. In other words, DBNs are stacks of RBMs.

RBMs are neural networks composed of two layers: visible units, and hidden units. The visible units receive input and produce output. RBMs differ from Boltzmann Machines (BM) in the following respect: only the neurons between visible and hidden layer are connected. There are no connection between the neurons in a given layer (visible or hidden). Both

[2]Reminder: A network is classified as a deep neural network if it has 2 or more hidden layers.

BMs and RBMs use neurons which are stochastic, i.e. these neurons output 1 with some probability(boltzmann distribution), and 0 otherwise.

It seems DBNs are mostly used for "hyperspectral" data, in which spectrum in various bands is recorded for each pixels. For example, the following two studies apply different DBN strategies to two datasets (Indian Pines and Pavia - details in Figure 3 and obtain the following results:

Type	*Range*	*Number of Bands*	*Kinds of objects*	*pixels*	*Spatial resolution*
Indian Pines	0.4-2.5μ m	224	16	144*144	20m
University of Pavia	0.43~0.86μ m	115	9	610*340	3.7m
Salinas	unclear	220	16	512*217	1.3m
Radarsat-2	0.43-0.9μ m	unclear	19	4952065	5.2-7.6m
UC Merced	unclear	unclear	21	256*256	0.3m
KSC	0.4-0.25μ m	224	13	512*614	18m

Figure 3: Datasets. Table from reference [13]

1. Chen et. al. [2]
   They proceed along two frameworks: spatial and spectral-spatial. In spectral-spatial framework, they combined spectral and spatial information to form a hybrid input. In spatial framework, they report OA of 93.20% on the Indian Pines dataset and 98.62% on Pavia dataset. In spectral-spatial framwork, they obtain OA of 95.95% on Indian Pines and 99.05 on Pavia.

2. Li et. al. [7]
   They obtain an OA of 95.81% on Indian Pines and an OA of 95.83% on Pavia.

2.2 Autoencoder

Autoencoders are artifical neural networks which learn a representation of input data, called coding, without supervision. These internal representations are forced to have lower dimensions than the original data, thereby forcing autoencoders to detect features. Finally, autoencoders generate random new data which resembles the input data (and is of same dimensionality as input data). Thus, autoencoders are composed of two parts: encoders - converting input to an internal representation, and decoders - converting internal representation to output. To ensure that the autoencoders do not simply learn an identity function constraints are enforced. These constraints force the network to find a way to learn the most important features. Usually, these architecture have logistic regression for fine-tuning and classification as their final layers.

1. Chen et. al. [1]
   As before, their work is segregated in two frameworks described above. They use an architecture based on Stacked Autoencoders (SAE). In spatial framework, they report Overall Accuracies (OA) of 97.76% on the KSC dataset and 98.12% on Pavia dataset. In spectral-spatial framwork, they obtain OA of 98.76% on KSC and 98.52 on Pavia. They also make interesting studies on the effects of depth of layers, number of neurons, quality of input reconstruction by the decoder, optimization of training time and comparison with other methods.

2. Xing et. al. [12]
   They use an architecture called stacked denoising autoencoders. They report OA of 92.06% on Indian Pines dataset and an OA of 95.97% on Pavia dataset.

2.3 Convolutional Neural Network

CNNs are one of the most popular variant of deep neural networks. Their popularity is, in part, due to the central role they play in driving the research in computer vision. This is easily seen through the annual results of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), starting from 2010. In only 5 years, the top-5 error rate for image classification fell from 26% to less than 3%. The top-5 error rate is a measure of how poor a system's prediction can be. It is the percent of test images for which the top-5 prediction from a system did not include the correct class/label. Some of the famous milestones in the evolution of CNN architecture are LeNet (1998), AlexNet (2012), GoogLeNet(2014), ResNet(2015), UNet (2015) along with the ideas of Inception, Xception modules and Residual units.

CNN architecture was inspired by the architecture of the Visual Cortex in Cats and Monkey (presumably also humans?) from the works of David Hubel and Torsten Wiesel (Nobel Prize in 1981). A typical CNN architecture is made up of several convolution layers intermixed with pooling layers followed by fully connected layers. Figure 4 shows one such architecture. Figure 5 explains the roles of convolution and pooling layers in a self-explanatory way.

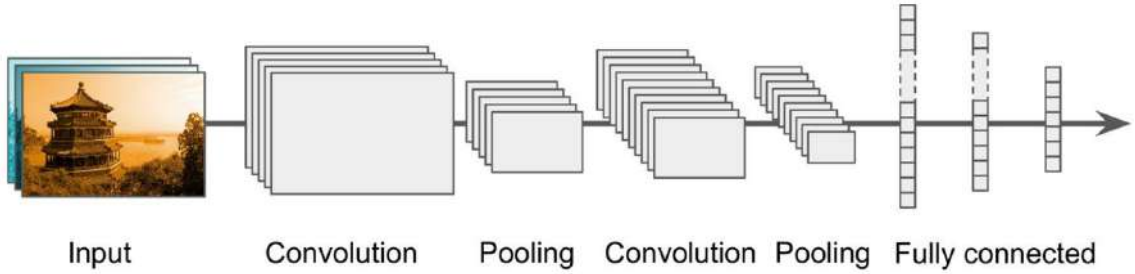

Figure 4: Typical CNN Architecture. From Geron's book.

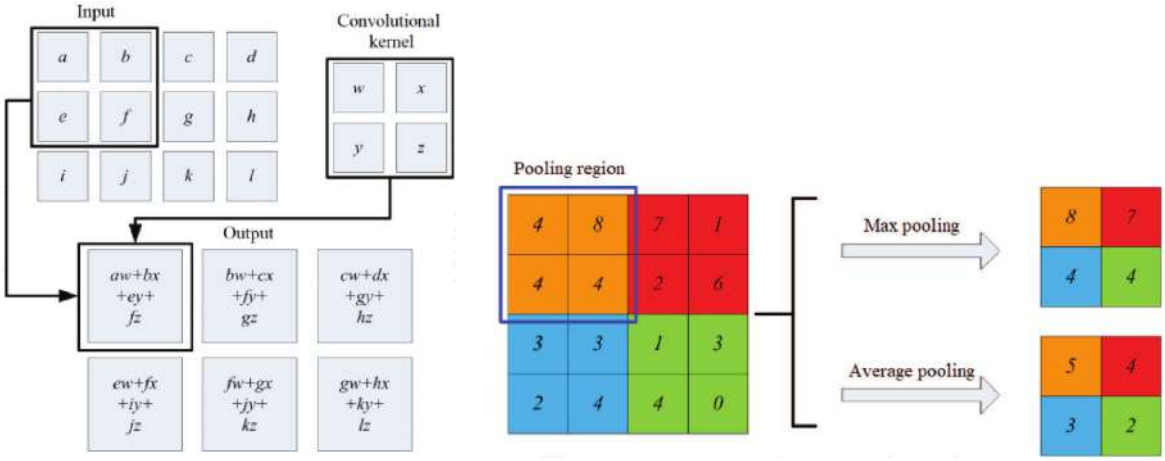

Figure 5: Function of Convolution and Pooling Layer from [8]

Application of CNNs towards LULC classification found a major boost with the advent of major competitions. Some of them are described in the next subsection. Before moving to the

next subsection, it is imperative to quote a study applying CNN on Indian Pines and Pavia dataset for comparison with DBNs and SAEs. Heming et. al. report OA of 97.45% on Indian Pines and an OV of 98.35% on Pavia dataset using sparse representation of hyperspectral images [6].

2.4 LULC Competitions, Datasets and Interesting Studies

1. fMOW Challenge

   The Intelligence Advanced Research Projects Activity (IARPA) is an organizational branch of the Office of the Director of National Intelligence, USA. IARPA hosted the Functional Map of the World (fMOW) challenge in 2017-18. The aim of this challenge was to foster breakthroughs in automated analysis of overhead imagery using ML/DL capabilities. The dataset consisted of 1,047,691 images from 207 countries along with metadata [3]. The competition recognized 63 different classes. Some of these classes are: airstrips, oil and gas facilities, mines, shipyards, ports, wind farms, office building, police station, crop field, solar farm, swimming pool, golf course, dam, hospital etc. Unique features of this dataset are: gigantic dataset (over 1 million instances), temporal information (to capture events like flooding, etc), metadata (allows for triangulation) etc. Some of the metadata features included - sun elevation, sun azimuth, target azimuth, off-nadir angle, cloud cover, time zone, time stamp etc.

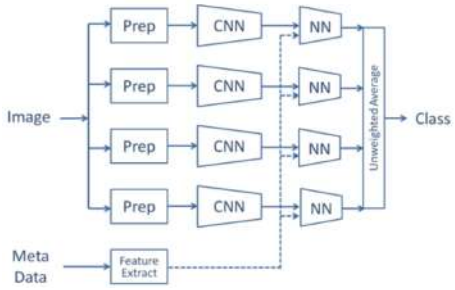

Figure 6: Ensemble CNN architecture from [11]

   Many of the winning entries used ensemble CNNs. For example, the entry by Pritt et. al. used the architecture depicted in Figure 6. Their entry, ranked at 5th position, had an OA of 83%. For 15 classes out of 63, the classification accuracy was over 95%.

2. EuroSAT - Sentinel-2 Satellite Images
   Helber et. al. apply deep CNN to EuroSAT dataset [5]. Their dataset has 27000 labelled images (64 by 64 pixels) with 10 classes: Industrial Buildings, Residential Buildings, Annual Crop, Permanent Crop, River, Sea & Lake, Herbaceous Vegetation, Highway, Pasture and Forest. The data consists of 13 bands (including rgb) which varies from 443 nm to 2190 nm in central wavelength.

   They report OA's from two classifiers: GoogLeNet 98.18% and ResNet-50 98,57%. Note - they manually discarded images with dead pixels due to clouds or snow.

3. DSTL - Kaggle Competition

   The Defence Science and Technology Laboratory (DSTL), UK hosted a competition for LULC classification from satellite imagery on Kaggle. The dataset consisted of 25 high-resolution images covering an area of 1 sqkm on ground. The task was to classify

10 kinds of objects: Buildings, Misc. manmade structures, roads, tracks, trees, crops, waterway, standing water, large vehicles, and small vehicles. The distribution of classes in the dataset was uneven (28% for crops to 0.8% for roads). Additionally, the images had three versions: grayscale, rgb, and 16 band multispectral.

The winning entry had a score of 0.49 Intersection over Union (IoU). An IoU of 0 means complete mismatch between predictions and ground truth and vive versa for IoU 1.

Deepsense.ai, a private firm ranked 4th with an IoU of 0.46. They employed a U-Net architecture on augmented data (combining all bands to form 20-channels per image). Figure 7 shows the architecture they used.

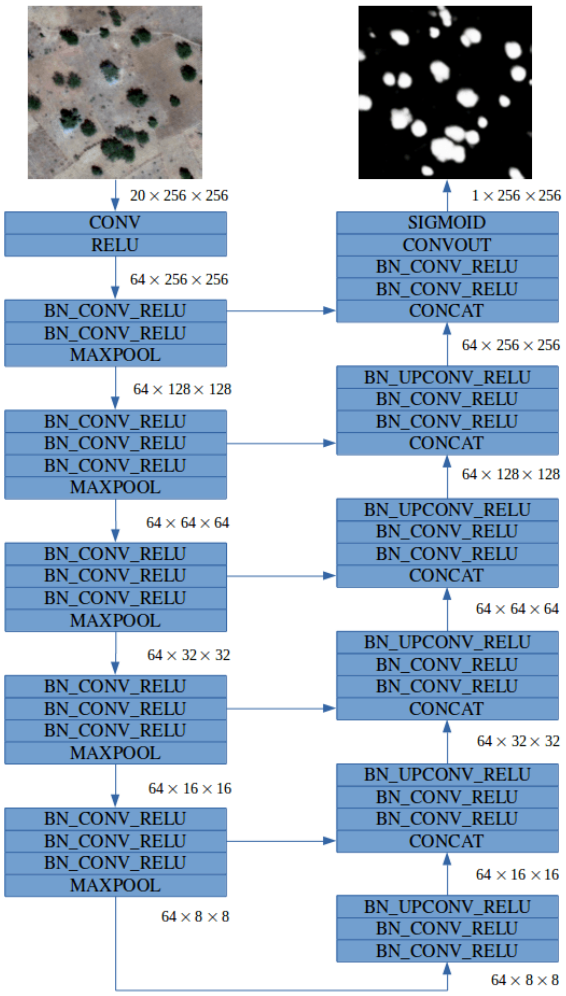

Figure 7: U-Net architecture from Deepsense.ai's blog.

4. Semantic Segmentation
   Marmanis et. al. design Fully Convoluted Networks to do image segmentation at the pixel level [9] with very high resolution images (10 cm ground sampling distance).

Although, the list above is far from completion, the author believes that it is enough to give a general idea of the breadth of variety and depth of accuracy that is offered by Deep Neural Networks for LULC classification.

3 Terrain Classification via Sequential Model

The various architectures described above are useful when high-resolution aerial images are available and the task at-hand requires identification of small objects like trees, roads, cards, buildings etc. Since, classification of terrains has only 5 label classes and does not currently care about object detection, a simpler architecture suffices. In this section, the network used for terrain classification and the results obtained are described.

Figure 2 already showed the 5 classes that are to be classified: Water, Mountains, Polar Caps, Forests and Deserts. The original images provided to the author were large images (1024x1024 pixels) and comprised of only one single terrain. As with any other data-science task, the first task for this project also involved data pre-processing. In pre-processing, composite images comprising of multiple terrains were created (512x512 pixels). The program picks a random point, then picks a random image from the given dataset (It was ensured that any one image from the original dataset is not used more than once for creating a composite image) and fills a pre-determined area around this point (in a circular manner). The area is determined by dividing the total area of the composite image by the number of different images that will be present in the final composite image. Some amount of randomness was added to the radius, determined by the last step, to ensure departures from exact circles. Also, since the starting point is picked randomly, considerable overlap is possible. The result of this procedure leads to a variety of composite images which can be seen in Figure 8. As an example, this figure shows composite images with 2, 4, 6, 8 and 10 terrains. However, for the purposes of further training composite images with 2, 4 and 6 terrains only were used. 150 images (50 each from 2, 4, and 6 terrains), created from the training set, became the training set and 90 images (30 each from 2, 4 and 6 terrains), created from the training set, comprised the testing set.

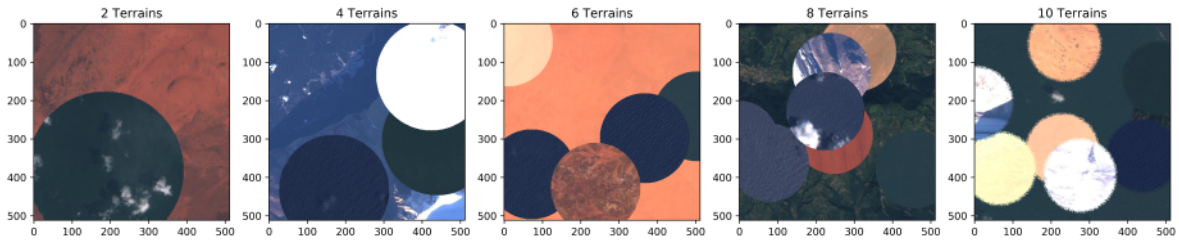

Figure 8: Composite images prepared from the original dataset.

A sequential model using the Keras API on tensorflow was created for this classification problem. A schematic sketch of this model is shown in Figure 9. There are 5 layers in this model with 64 neurons each. The first layer receives the input and the last layer produces the probabilistic output for each class. The first four layers have RELU (rectified linear unit) activation, while the last layer is softmax activated. Categorical cross-entropy is used to calculate the loss at each step and Adam is used as an optimizer. There are 13,061 trainable parameters in this model.

From the 150 instances in the training set, 120 are used for training and 30 are used for validation. Run 1 of training uses a learning rate of 0.001, and comprises of 100 epochs. At the end of run-1, this model has a 94.08% accuracy on the training set and 92.22% accuracy on the validation set. Figure 10 plots the learning curve from run 1. Since, the slope in the learning curve isn't completely flat, the model is further trained. This time the learning rate is set at 0.0001 and 50 epochs. Run-2 gives an accuracy of 94.62% on the training set and 93.42% on the validation set. Figure 11 shows the learning curve for run 2. Finally, to be

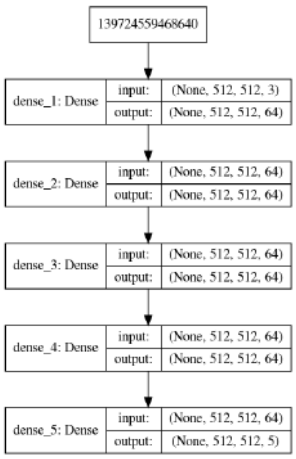

Figure 9: Sequential model used for Terrain Classification. The model has 3 hidden layers.

completely sure, another round of training is used. This time the learning rate is further lowered to 0.00001 for 50 epochs. Run 3 gives an accuracy of 95.62% on training and 93.91% on the validation set. Figure 12 shows the learning curve for run 3. Comparing the learning curve from Figures 11 and 12, the smoothness of the latter curve can be attributed to the lowered learning rate. Additionally, all runs lead to a higher accuracy on the training set as compared to the validation set. That this feature was not a particular effect of one possible run, this entire training (all 3 runs) was re-run again, and this effect still persisted. This could be attributed to the larger training set.

Figure 10: Learning curve for training in run 1.

Finally, this model was tested against the 90 images in the test set. The OA is at 95.3035%. Figure 16 shows some of the test images, their ground truth and predictions. For visualization, the 5 classes are colored in the ground truth and prediction images. Orange color depicts deserts, blue is for water, white for polar caps, green for forests and grey for mountains. As can be seen from the figure, the model is generally quite good. To estimate and quantify the errors, normalized confusion matrix is calculated. Figure 13 shows the confusion matrix for the predictions. The y-axis of each matrix corresponds to the true label

Figure 11: Learning curve for training in run 2.

Figure 12: Learning curve for training in run 3.

and the x-axis corresponds to the predicted label. In an ideal scenario, this matrix should be equal to the identity matrix. And non-zero element in the confusion matrix indicates an error in the predictions. From the matrix on the left, it is clear that all classes except mountains have an accuracy of over 95%. The matrix on the right highlights the errors. Deeper shades of color indicate larger error percentages. Most errors consist of mounatins being classified as forests and water bodies.

Additionally, as a test on the model, images composed of only 1 pixels were generated and the entire procedure described above was repeated. After 1000 epochs at a learning rate of 0.001, the training accuracy reached 95.83% and the validation accuracy reached 96.67%. This model performed rather poorly on the test set and achieved an overall accuracy of 85.5556%. This means that the model is overfitting, which is expected since the size of each instance was severely reduced (from 512x512 pixels to 1x1 pixels). Figure 14 shows the learning curve and Figure 15 shows the normalized confusion matrix from the prediction of the 1x1 model. The confusion matrices show that the errors in 1 pixel image are similar to

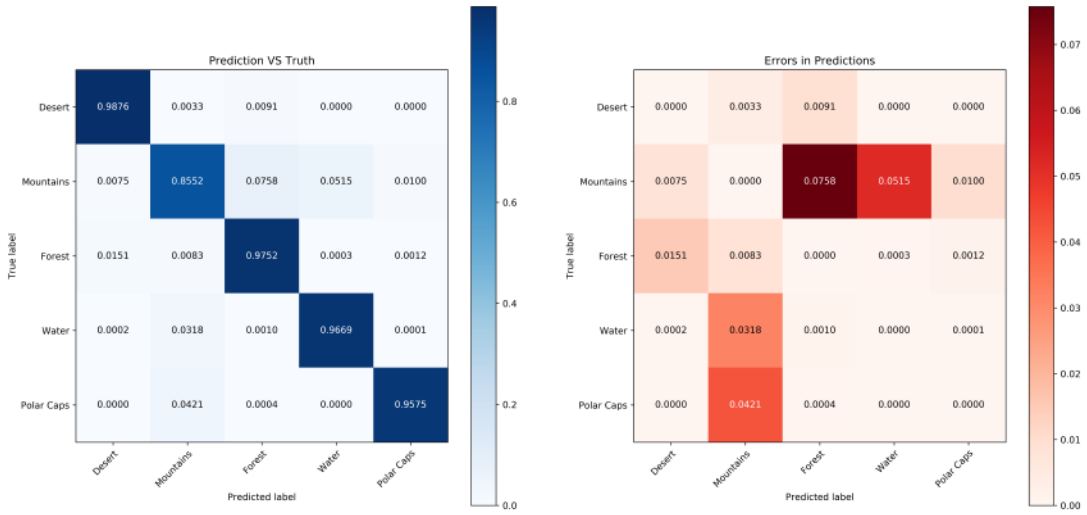

Figure 13: Normalized confusion matrix for the predictions from the model.

the errors produces in the 512x512 images. This suggests that this model is not using any features that maybe spread over several pixels but only features present in one pixel, the rgb values of the pixel.

Figure 14: Learning curve for the same model applied to 1 pixel images.

Further improvements in this model could be obtained by using a deeper network or by trying one of the architectures described in section 2. Additionally, demanding more classes might require the use of mode advanced strategies. To mitigate errors, one might use classes which are exclusive than forests and mountains.

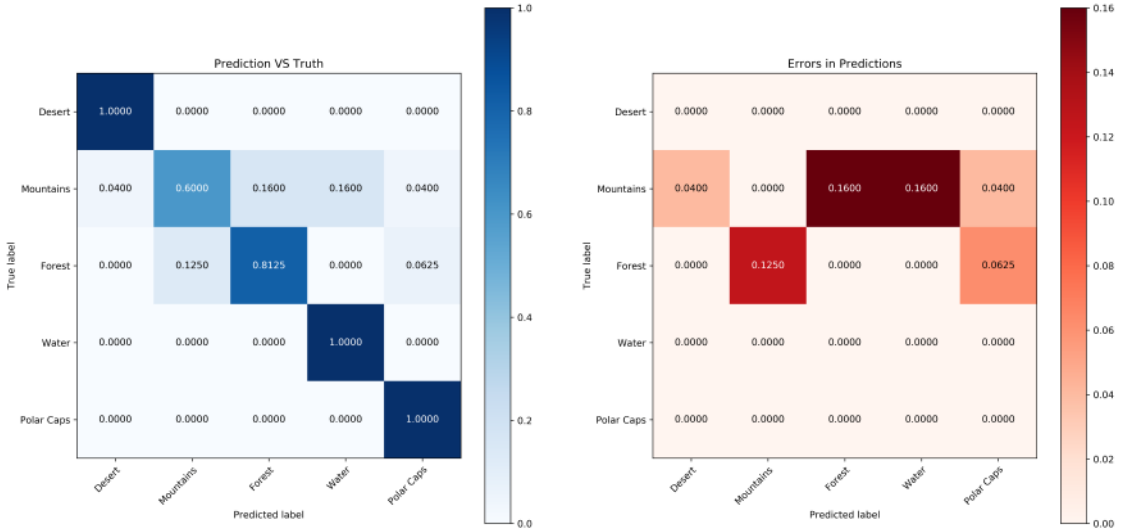

Figure 15: Normalized confusion matrix from predictions generated from 1 pixel images.

References

[1] Yushi Chen, Zhouhan Lin, Xing Zhao, Gang Wang, and Yanfeng Gu. Deep learning-based classification of hyperspectral data. *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of*, 7:2094–2107, 06 2014.

[2] Yushi Chen, Xing Zhao, and Xiuping Jia. Spectralspatial classification of hyperspectral data based on deep belief network. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8:1–12, 06 2015.

[3] Gordon Christie, Neil Fendley, James Wilson, and Ryan Mukherjee. Functional map of the world. In *CVPR*, 2018.

[4] Antonio Di Gregorio, Matieu Henry, Emily Donegan, Yelena Finegold, John Latham, Inge Jonckheere, and Renato Cumani. Land cover clsasification system. *Food And Agriculture Organization of the United Nations*, 2016. http://www.fao.org/3/a-i5232e.pdf.

[5] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *CoRR*, abs/1709.00029, 2017.

[6] Liang Heming and Qi Li. Hyperspectral imagery classification using sparse representations of convolutional neural network features. *Remote Sensing*, 8:99, 01 2016.

[7] Chenming Li, Yongchang Wang, Xiaoke Zhang, Hongmin Gao, Yao Yang, and Jiawei Wang. Deep belief network for spectralspatial classification of hyperspectral remote sensor data". *Sensors (Basel) MDPI*, 19, 2019.

[8] Chang Luo, Hanqiao Huang, Yong Wang, and Shiqiang Wang. *Utilization of Deep Convolutional Neural Networks for Remote Sensing Scenes Classification*. 11 2018.

[9] D. Marmanis, J.D. Wegner, S. Galliani, K. Schindler, M. Datcu, and U. Stilla. Semantic segmentation of aerial images with an ensemble of cnns. In *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol-

ume 3, 2016. Available at: https://www.isprs-ann-photogramm-remote-sens-spatial-inf-sci.net/III-3/473/2016/isprs-annals-III-3-473-2016.pdf.

[10] Otávio AB Penatti, Keiller Nogueira, and Jefersson A Dos Santos. Do deep features generalize from everyday objects to remote sensing and aerial scenes domains? In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 44–51, 2015.

[11] Mark Pritt and Gary Chern. Satellite image classification with deep learning. *IEEE Applied Imagery Pattern Recognition Workshop (AIPR), 2017*. https://ieeexplore.ieee.org/document/8457969.

[12] Chen Xing, Li Ma, and Xiaoquan Yang. Stacked denoise autoencoder based feature extraction and classification for hyperspectral images. *Journal of Sensors*, 2016:1–10, 01 2016.

[13] Chuchu Yao, Xianxian Luo, Yudan Zhao, Wei Zeng, and Xiaoyu Chen. A review on image classification of remote sensing using deep learning. pages 1947–1955, 12 2017.

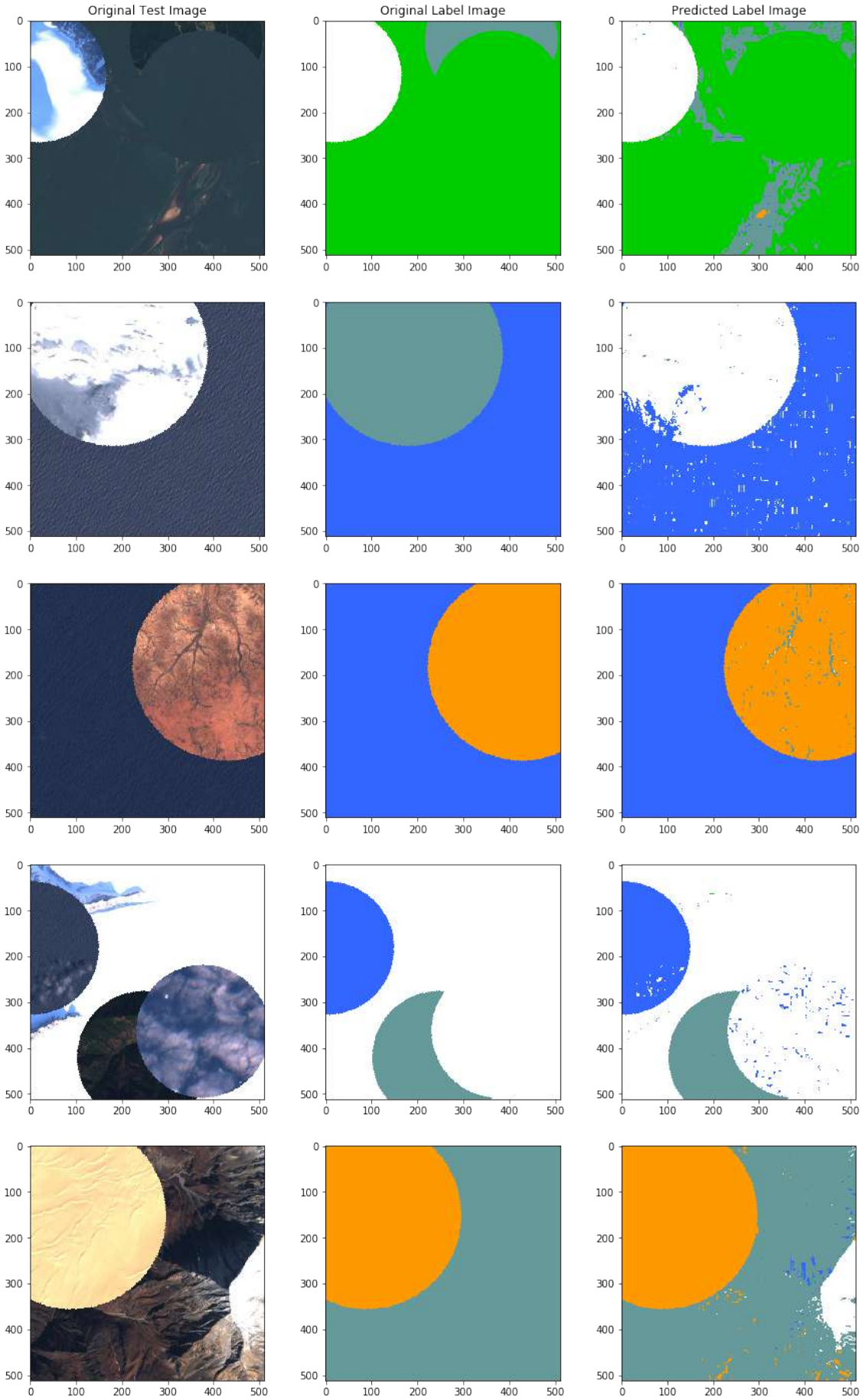

Figure 16: Comparison of the predictions from the model with ground truth.