

Classifying Land-Cover Using Texture Statistics

Travis Askham, Advisor: Todd Wittman
Completed as Part of a Math Research 199 at UCLA

January 12, 2010

Abstract

We present a method for labeling the land-cover in an aerial image using image texture statistics and a trained neural network to interpret the texture data. The introduction includes a brief discussion of land-cover mapping and the types of land-cover classes we would like to identify. There is a section describing the texture statistics used and the Gray Level Co-occurrence Matrix (GLCM) for calculating them. There is also a section describing neural networks and the specific parameters used for our neural networks. The results section contains the images used for training the neural network, example images for the texture statistics, and labeled aerial images.

1 Introduction

In remote sensing, it is common to segment an aerial image into land-cover classes. The classified image can then be used on its own or applied to other algorithms. One intended use of the results presented here is their application to modeling the relative probability of events, such as burglaries, within a spatial region. Work is currently being done [1] to incorporate geographic data into these models for more accurate results. Often the geographic information comes from satellite imagery which is then hand-labeled. The process of hand-labeling these images is time consuming and an automated method for labeling is desirable.

When classifying the regions of an image there are several factors to consider: the number and type of classes to identify, the information used to determine which class a pixel belongs to, and how to translate that information into a pixel-wise classification. We chose ten classes for our classification: low density residential/commercial, high density residential/commercial, industrial, roads, parks/forest, bare soil, desert/sand, crops/grass, calm water, and rough water. We calculated image texture features at each pixel using Haralick's Gray-Level Co-occurrence Matrix. Combinations of these texture features were found to be an accurate method for classifying aerial images in [2]. To this end, we combined the texture statistics using a neural network which we trained on hand-labeled data.

2 Texture Statistics

2.1 The Gray Level Co-occurrence Matrix

The Gray Level Co-occurrence Matrix (GLCM), as described in [3], is a tool for calculating numerous texture statistics. The GLCM describes how frequently two gray levels appear next to one another within an image. There is a separate matrix for gray levels occurring to the right and left of one another (horizontal), up and down from one another (vertical), on the up-right diagonal

from one another, and on the up-left diagonal from one another. To populate a GLCM, you start with a square matrix of zeroes with side length equal to the number of gray-levels in the image. Then you go through the image and for every pair of pixels that satisfies the spatial relation for this GLCM (e.g. pixels to the right and left of each other) with respective intensities (gray values) i and j you increment the value of the GLCM in the i th row and j th column by one. For example, the 4-by-4 pixel image below with 4 gray levels (0-3) has a horizontal GLCM, P_H , which looks like:

$$\text{Image} = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 2 & 2 & 2 \\ 2 & 2 & 3 & 3 \end{pmatrix} \text{ and } P_H = \begin{pmatrix} 4 & 2 & 1 & 0 \\ 2 & 4 & 0 & 0 \\ 1 & 0 & 6 & 1 \\ 0 & 0 & 1 & 2 \end{pmatrix}$$

In this example from [3] it is important to note that the GLCM P_H is symmetric, which makes some calculations for the texture statistics faster, and that it is not normalized at this point. When calculating the image texture statistics, any GLCM P is treated as a probability distribution and is therefore normalized by dividing each entry by the sum of the entire matrix. The value of the sum over the whole P matrix depends only upon which spatial relationship the matrix represents and the number of columns and rows in the image. For example, when considering horizontal pairs in a N_r -by- N_c image there are $2(N_c - 1)$ pairs per row and N_r rows giving a total of $2N_r(N_c - 1)$ horizontal pairs. Once the GLCMs (P_H, P_V, P_{UR}, P_{UL}) are calculated and normalized, the texture statistics are then derived from these probability distributions. To achieve rotation invariance in the texture statistics, the values derived from the P_H, P_V, P_{UR} , and P_{UL} matrices are averaged.

For our statistics, we first converted the RGB images to gray-scale and then quantized them to 8 gray levels (this value was chosen by visual inspection). We then used a localized version of the process described above by taking a small window (11-by-11) around each pixel and calculating the GLCMs and statistics over this window to assign texture statistic values on a per-pixel basis.

2.2 Notation

The following is some notation used in the formulas for the image texture statistics.

- $p(i, j) := (i, j)$ th entry in a normalized GLCM (probability density)
- $p_x(i) := \sum_{j=1}^{N_g} p(i, j)$
- $p_y(j) := \sum_{i=1}^{N_g} p(i, j)$
- $p_{x+y}(k) := \sum_{i,j:i+j=k} p(i, j)$ for $k = 2, 3, \dots, 2N_g$
- $p_{x-y}(k) := \sum_{i,j:|i-j|=k} p(i, j)$ for $k = 0, 1, \dots, N_g - 1$
- $\mu :=$ mean of $p(i, j)$
- $\mu_x, \mu_y :=$ means of p_x and p_y respectively
- $\mu_{x-y} = \sum_{i=0}^{N_g-1} i p_{x-y}(i)$
- $\sigma_x, \sigma_y :=$ standard deviations of p_x and p_y respectively
- $HX, HY :=$ the entropies of p_x and p_y respectively

- $HXY1 := - \sum_i \sum_j p(i, j) \log(p_x(i)p_y(j))$
- $HXY2 := - \sum_i \sum_j p_x(i)p_y(j) \log(p_x(i)p_y(j))$
- $N_g :=$ the number of gray levels

2.3 The Statistics

The following are the formulas used to calculate thirteen of the texture statistics from [3].

1. Angular Second Moment

$$f_1 = \sum_i \sum_j p(i, j)^2$$

2. Contrast

$$f_2 = \sum_{n=0}^{N_g-1} n^2 p_{x-y}(n)$$

3. Correlation

$$f_3 = \frac{\sum_i \sum_j (ij) p(i, j) - \mu_x \mu_y}{\sigma_x \sigma_y}$$

4. Sums of Squares: Variance

$$f_4 = \sum_i \sum_j (i - \mu)^2 p(i, j)$$

5. Inverse Difference Moment

$$f_5 = \sum_i \sum_j \frac{1}{1+(i-j)^2} p(i, j)$$

6. Sum Average

$$f_6 = \sum_{i=2}^{2N_g} i p_{x+y}(i)$$

7. Sum Variance

$$f_7 = \sum_{i=2}^{2N_g} (i - f_6)^2 p_{x+y}(i)$$

8. Sum Entropy

$$f_8 = - \sum_{i=2}^{2N_g} p_{x+y}(i) \log(p_{x+y}(i))$$

9. Entropy

$$f_9 = - \sum_i \sum_j p(i, j) \log(p(i, j))$$

10. Difference Variance

$$f_{10} = \sum_{i=0}^{N_g-1} (i - \mu_{x-y})^2 p_{x-y}(i)$$

11. Difference Entropy

$$f_{11} = - \sum_{i=0}^{N_g-1} p_{x-y}(i) \log(p_{x-y}(i))$$

12. Information Measure of Correlation 1

$$f_{12} = \frac{f_9 - HXY1}{\max\{HX, HY\}}$$

13. Information Measure of Correlation 2

$$f_{13} = (1 - \exp[-2.0(HXY2 - f_9)])^{\frac{1}{2}}$$

3 Neural Networks

A neural network is a tool for information processing that is modeled after some of the processes in the brain. It is made up of simple elements, called neurons, which transmit signals to one another. There are connection links which send these signals from neuron to neuron and they each have an associated weight. Within the neuron, the sum of all of the weighted inputs is passed to an activation function which determines whether a neuron fires and with what strength. There are always input neurons and output neurons and sometimes there's a hidden layer of neurons in between.

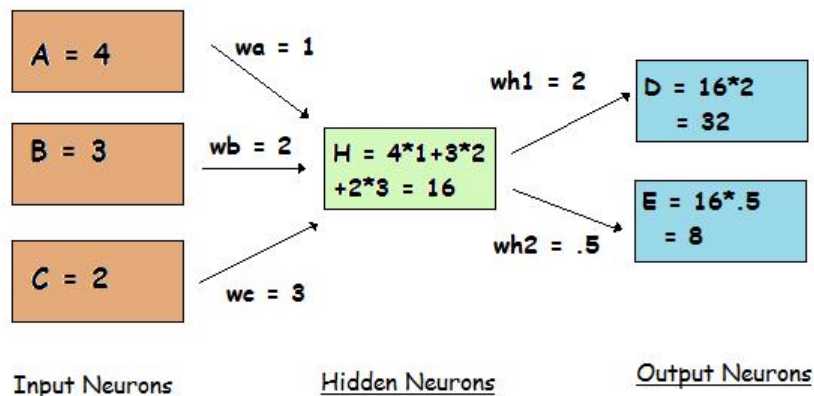


Figure 1: Diagram of a simple neural network running on a specific input, with the identity for its activation functions.

As seen in the image above, each neuron passes on its value and the connection links apply a weight to this value. The receiving neuron then gets the value of its activation function (the identity in the example) applied to the sum of the weighted inputs. For more about neural networks see [4].

When working with neural networks, there are two stages. The first is training. To train, you supply the network with sample input values and desired output values. The network then tries to “learn” appropriate weights for the connection links to give the desired output for each sample input. Once trained, the weights are set. Then in the second stage, you feed data through the neural network and see what output the network gives. For our neural network, we used the neural network toolbox in MatlabTM. We used the values of our 13 texture statistics as input neurons, a hidden layer of 20 neurons, and 10 output neurons, each corresponding to a different type of land-cover (a value of 1 being a positive for that specific class and a value of 0 being a negative). To train, we labeled regions of a few images by hand. These regions of the image were then converted to sets of sample input data (13 texture statistics) and target output data (vectors of 0’s and a single 1). These input and output pairs were then given to the MatlabTM neural network training function to assign the weights in the final neural network. We then passed the texture statistics for entire images to the trained neural network and converted the output to labeled images. Further, we trained a second neural network with the texture statistics plus the RGB values of the original image as inputs (a total of 16 input neurons) and created labeled images with this network as well. The idea is that neither of these networks needs further hand labeled data to classify images.

4 Results

4.1 Statistics Examples

The following are images which represent the values of the texture statistics calculated at each pixel in an image. The red areas correspond to large values of the texture statistic, while blue corresponds to small values.

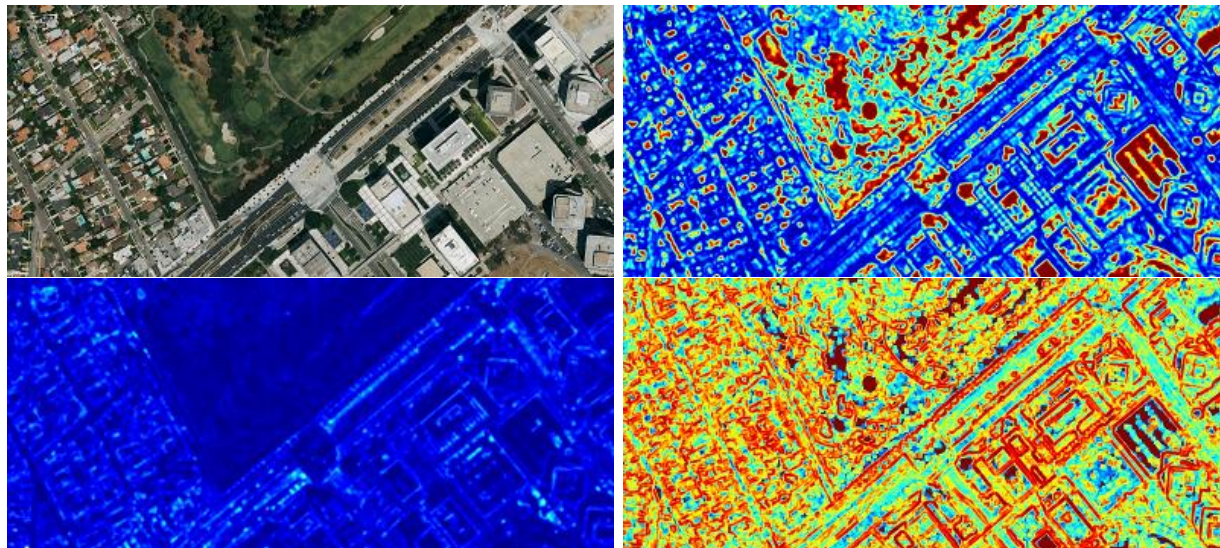


Figure 2: Example images for the texture statistics applied to a scene with low and high density residential/commercial areas and a golf course. Top left: original. Top right: Angular Second Moment (Homogeneity). Bottom left: Contrast. Bottom right: Correlation.

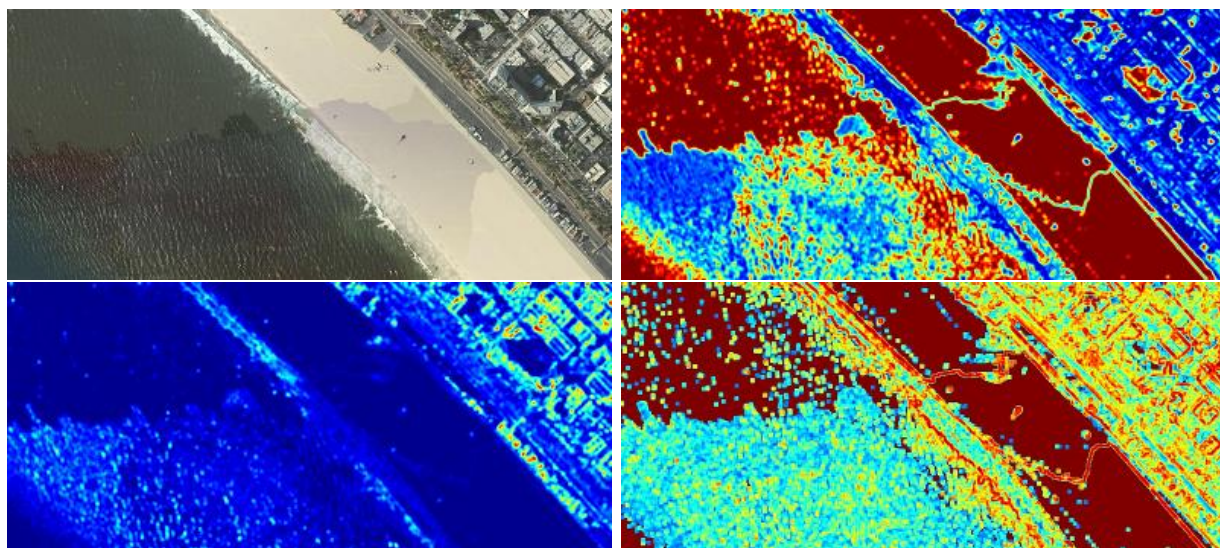


Figure 3: Example images for the texture statistics applied to a beach scene with both low and high density residential/commercial areas, sand, and rough water. Top left: original. Top right: Angular Second Moment (Homogeneity). Bottom left: Contrast. Bottom right: Correlation.

4.2 Training Data

The following are the original images and the sparsely labeled “ground truth” images that were used to train the neural network.

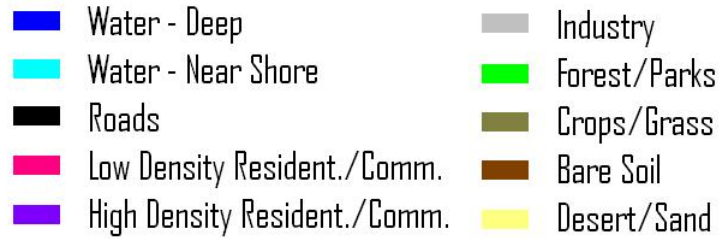


Figure 4: The legend for the land-cover categories.



Figure 5: Scene with both high and low density residential/commercial areas, as well as a golf course.



Figure 6: Scene with high density commercial areas and a large road in the middle.



Figure 7: Scene of low density residential housing.

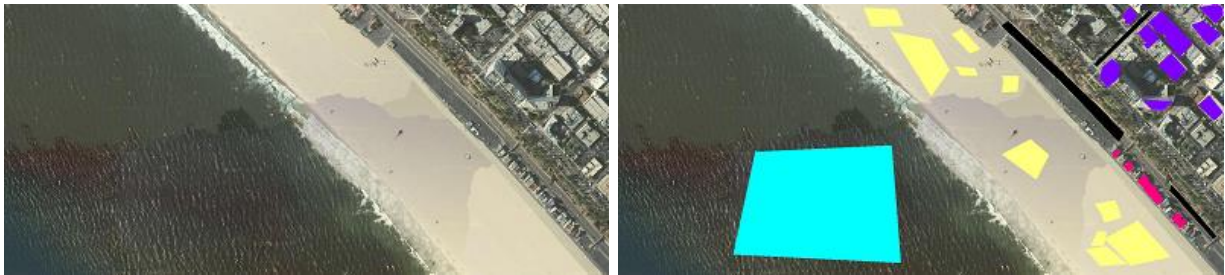


Figure 8: Scene of a beach and nearby low and high density residential/commercial areas.



Figure 9: Scene of an industrial area.

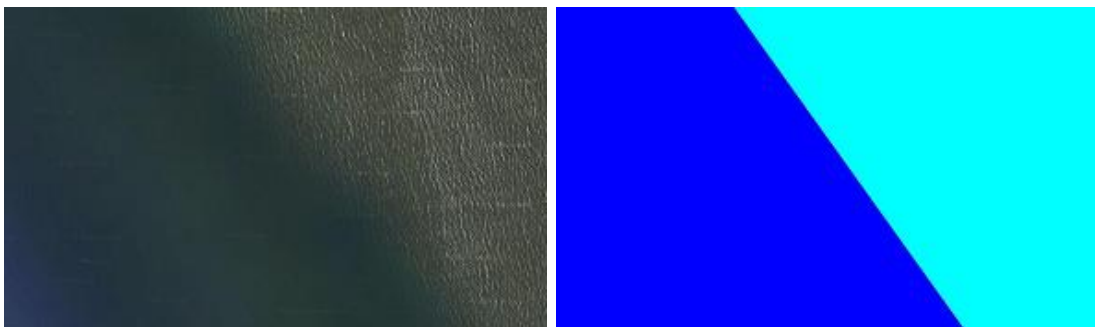


Figure 10: Scene of both smooth and rough water.



Figure 11: Scene of a rural area with crops and some houses.

4.3 Classification Results

The following are the results of classifying the images based on the texture statistics and running these through the trained neural network. The results are shown for both the trials where the RGB values were included as statistics and the trials where they were not included.

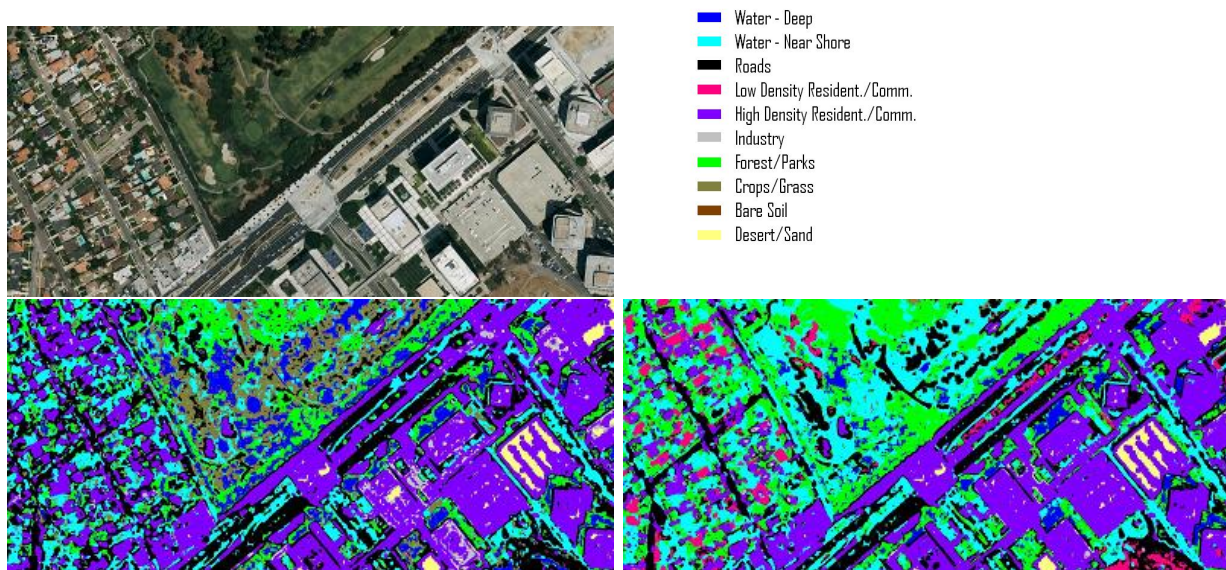


Figure 12: Results for the entire first training image. Top left: original. Top right: legend. Bottom left: classification without RGB values. Bottom right: classification with RGB values.

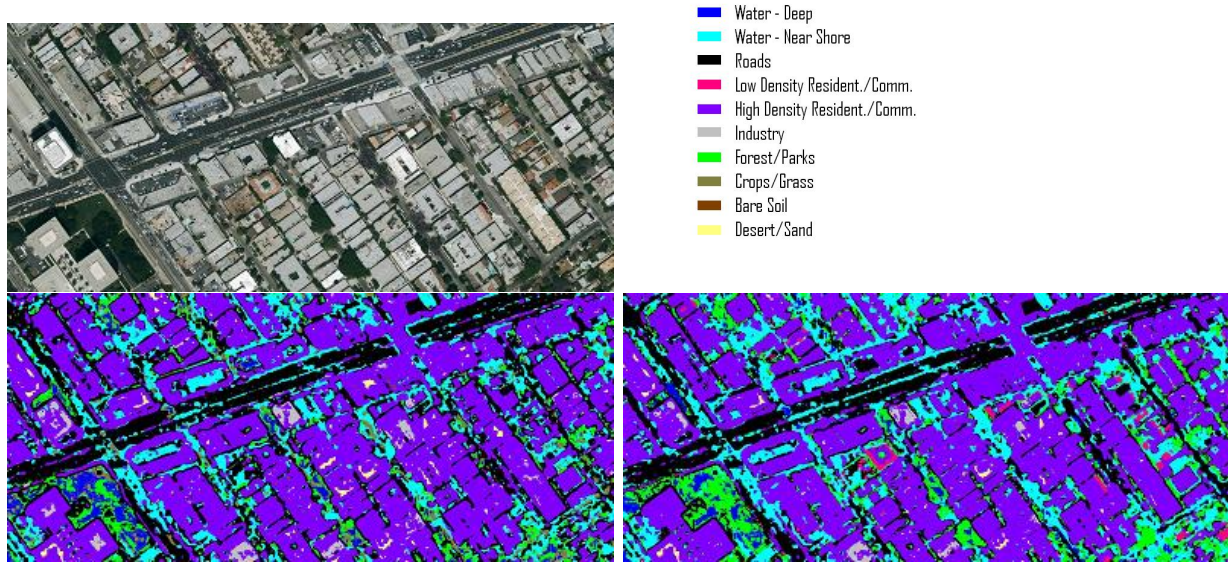


Figure 13: Results for the entire second training image. Top left: original. Top right: legend. Bottom left: classification without RGB values. Bottom right: classification with RGB values.

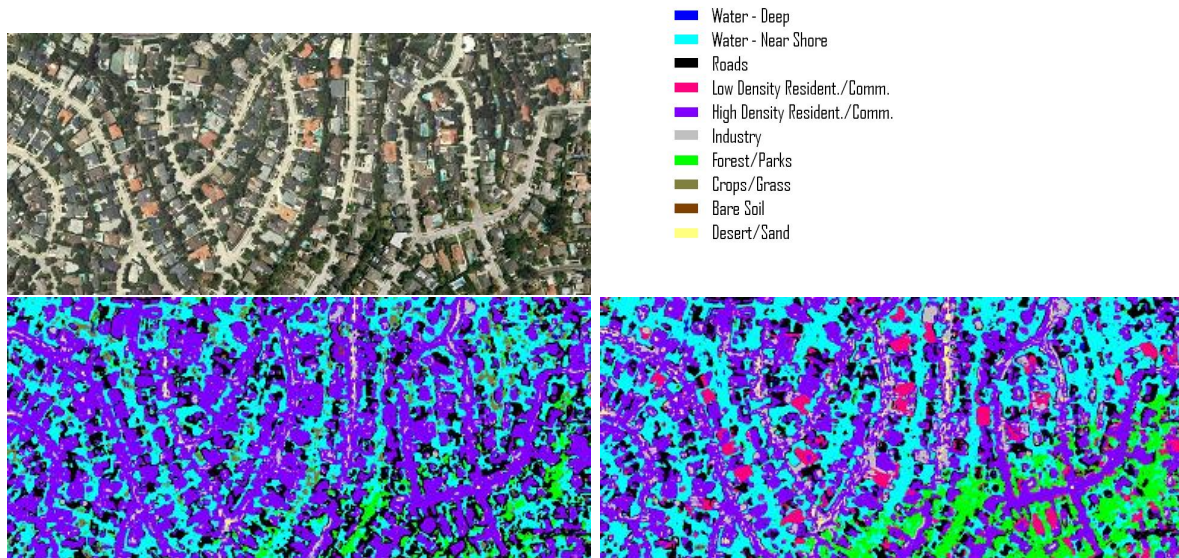


Figure 14: Results for the entire third training image. Top left: original. Top right: legend. Bottom left: classification without RGB values. Bottom right: classification with RGB values.

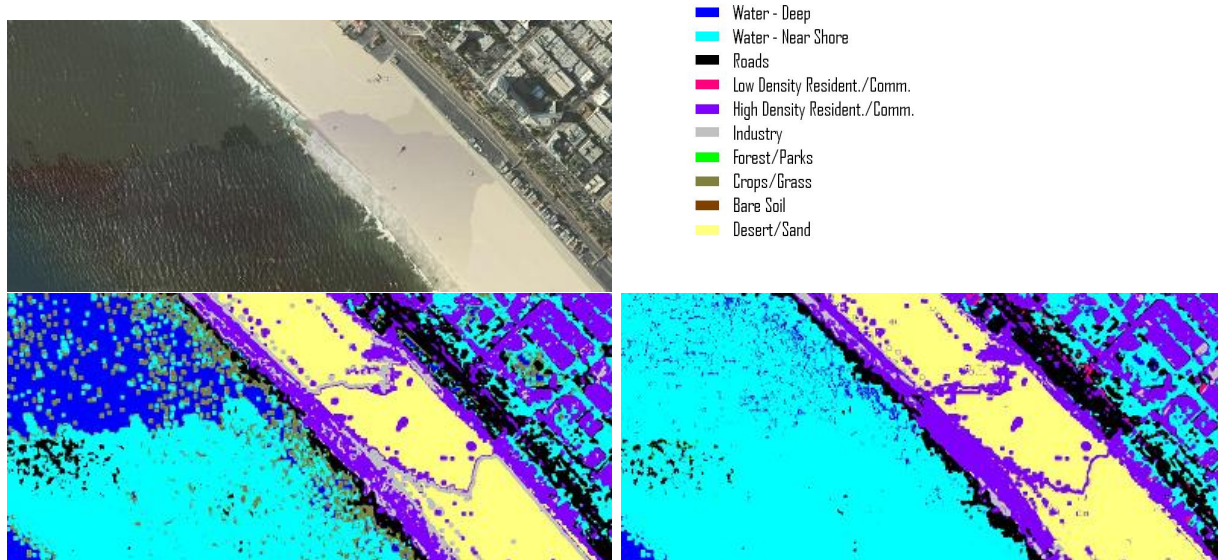


Figure 15: Results for the entire fourth training image. Top left: original. Top right: legend. Bottom left: classification without RGB values. Bottom right: classification with RGB values.

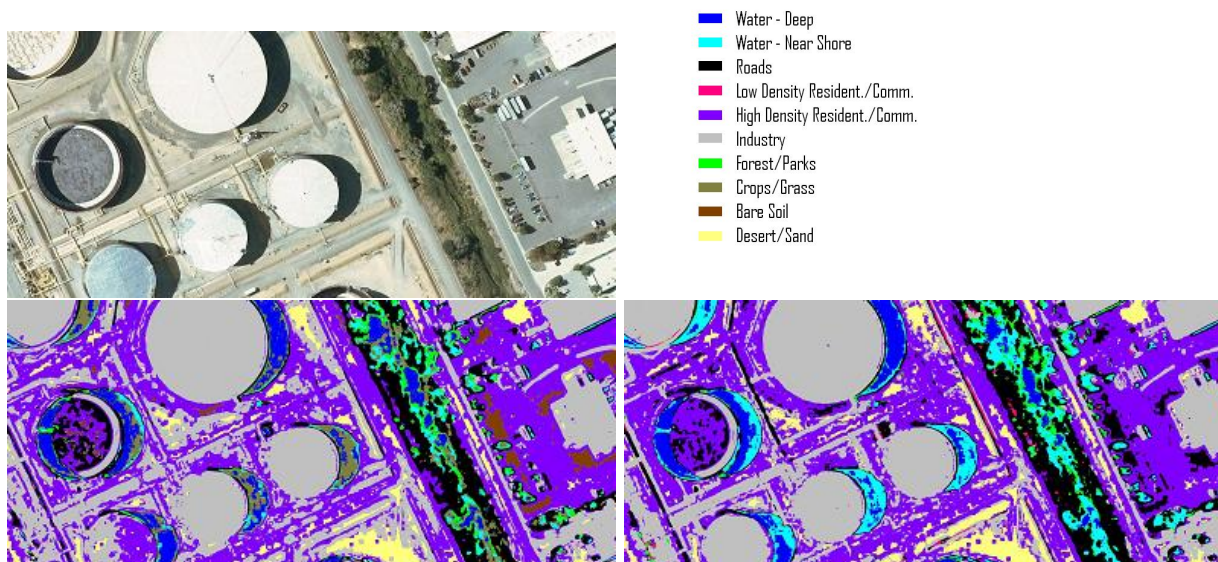


Figure 16: Results for the entire fifth training image. Top left: original. Top right: legend. Bottom left: classification without RGB values. Bottom right: classification with RGB values.

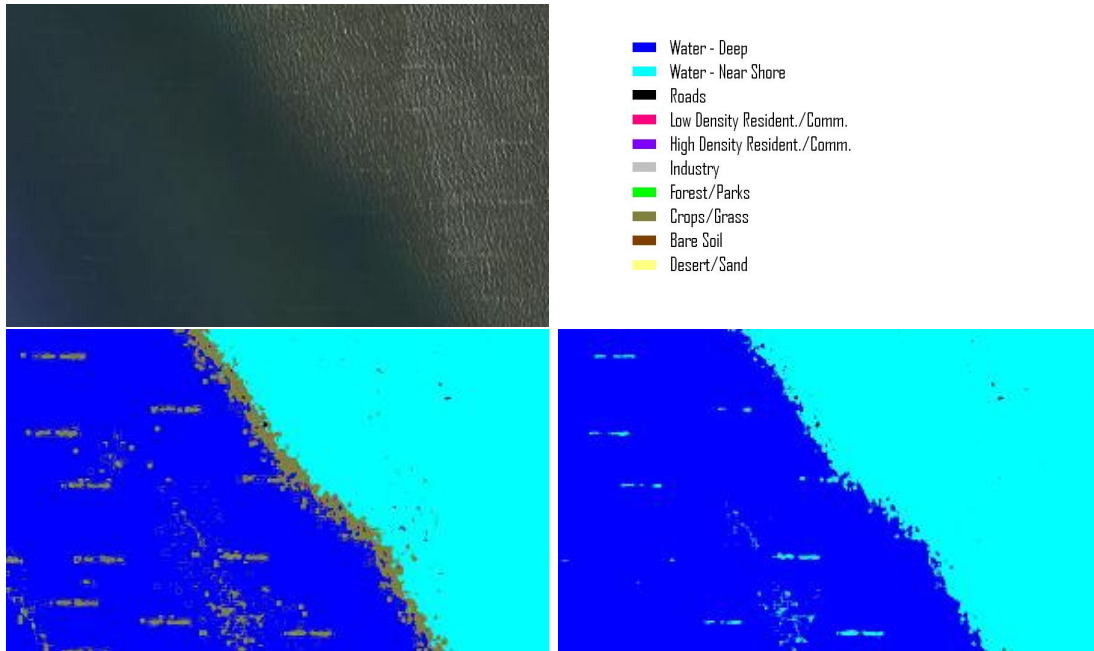


Figure 17: Results for the entire sixth training image. Top left: original. Top right: legend. Bottom left: classification without RGB values. Bottom right: classification with RGB values.

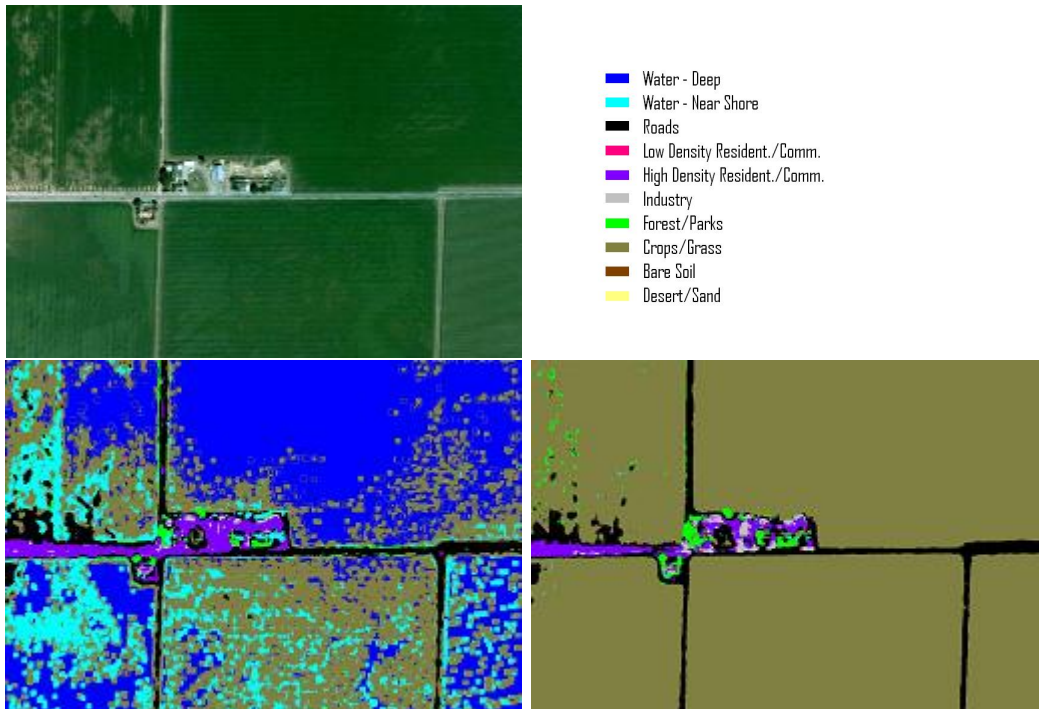


Figure 18: Results for the entire seventh training image. Top left: original. Top right: legend. Bottom left: classification without RGB values. Bottom right: classification with RGB values.

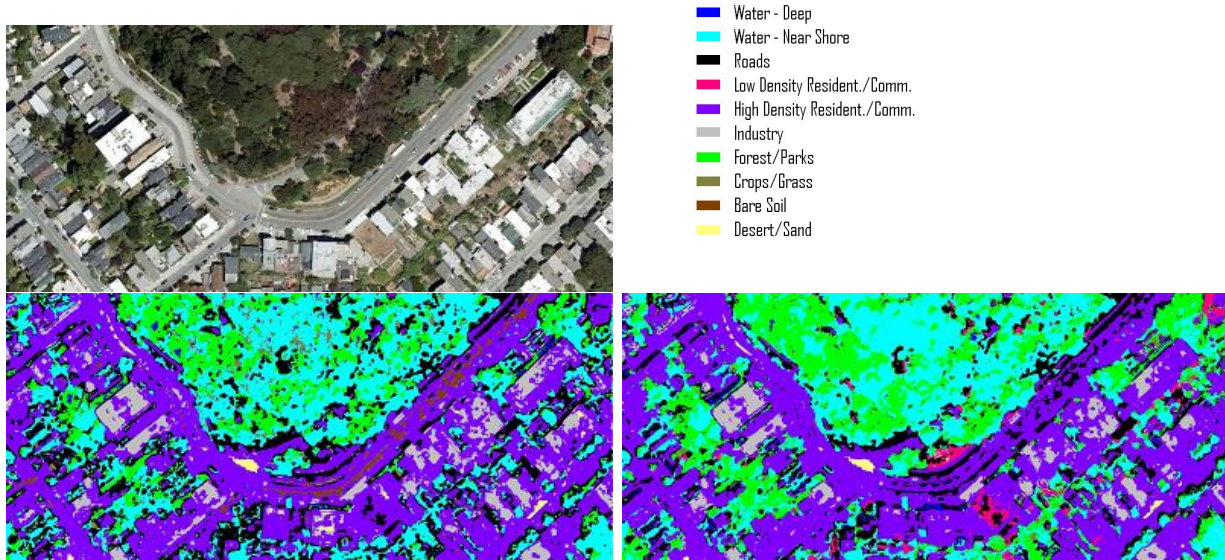


Figure 19: Results for the eighth image; no part of this image was used for training. Top left: original. Top right: legend. Bottom left: classification without RGB values. Bottom right: classification with RGB values.

5 Conclusions and Further Research

Although more quantitative results are still needed; by visual inspection of the result images, some of the strengths and weaknesses of the method are apparent. It tends to over-classify regions as water and nearly all commercial and residential areas are labeled as high density. Road materials other than dark asphalt tend to be classified as residential/commercial or water. As a positive, the method rarely misclassifies a natural land-cover for a man made land-cover and vice-versa. In light of these results, it seems that it would be beneficial to combine some classes, for example low and high density residential/commercial, to help simplify the neural network and overall classification process. Also, a few of the texture statistics seem too noisy to be of use and others seem redundant. By eliminating some of the texture statistics which contribute less to, or even confuse, the method, we could speed up both the process of calculating the texture statistics and the training of the neural network. Also, there appeared to be some improvement in the results where the RGB values were included as extra statistics; however, RGB values are affected by too many factors like shadows and time of day for them to be considered reliable. Thus, it may be useful to explore other color spaces, such as the Hue Saturation Value (HSV) color space, to be included as extra statistics. Finally, more diverse training data (e.g. with different lighting conditions, plant life, etc.) would allow the network to be more robust when applied to new images.

References

- [1] L.M. Smith, M.S. Keegan, T. Wittman, G.O. Mohler, A. Bertozzi, “Improving Density Estimation by Incorporating Spatial Information,” 2009.
- [2] L.P. Abeigne Ella, F. van den Bergh, B.J. van Wyk, M.A. van Wyk, “A Comparison of Texture Feature Algorithms for Urban Settlement Classification,” *IEEE IGARSS*, 2008.

- [3] R.M. Haralick, I. Dinstein, and K. Shanmugam, “Textural Features for Image Classification,” *IEEE Transaction on Systems, Man, and Cybernetics*, vol.3, pp. 610–621, 1973.
- [4] L. Fausett, *Fundamentals of Neural Networks*, Prentice Hall, Upper Saddle River, NJ, 1994.