MULTISPECTRAL IMAGE ANALYSIS USING RANDOM FOREST

Barrett Lowe and Arun Kulkarni

Department of Computer Science, The University of Texas at Tyler

ABSTRACT

Classical methods for classification of pixels in multispectral images include supervised classifiers such as the maximum-likelihood classifier, neural network classifiers, fuzzy neural networks, support vector machines, and decision trees. Recently, there has been an increase of interest in ensemble learning – a method that generates many classifiers and aggregates their results. Breiman proposed Random Forestin 2001 for classification and clustering. Random Forest grows many decision trees for classification. To classify a new object, the input vector is run through each decision tree in the forest. Each tree gives a classification. The forest chooses the classification having the most votes. Random Forest provides a robust algorithm for classifying large datasets. The potential of Random Forest is not been explored in analyzing multispectral satellite images. To evaluate the performance of Random Forest, we classified multispectral images using various classifiers such as the maximum likelihood classifier, neural network, support vector machine (SVM), and Random Forest and compare their results.

KEYWORDS

Classification, Decision Trees, Random Forest, Multispectral Images

1. INTRODUCTION

Remotely sensed data provides a wealth of information. Remote sensing satellites such as Landsat provide multispectral images of earth's surfaces. The technique of remote sensing relies, to a great extent, on interaction of electromagnetic radiation with matter. The remotely measured signal expressed as a function of a wavelength is referred to as a spectral signature of the target object on which measurements have been made. In principle spectral signatures are unique, that is different objects have different spectral signatures. It is therefore possible to identify an object from its spectral signature. Multispectral images are used in many applications such as land use mapping, agriculture, water resources, and military reconnaissance. Landsat remote sensing began in earnest in with launching of the first remote sensing satellite in 1972. Today Landsat-8 is orbiting the earth. The Operational Land Imager (OLI) sensor on Landsat payload provides images in nine spectral bands.

Multispectral images are oftenanalyzed using conventional statistical methods, and soft computing techniques such as neural networks, fuzzy inference systems and fuzzy neural systems. Conventional methods employed for classifying pixels in multispectral images include the maximum likelihood classifier, the minimum distance classifier, and various clustering techniques such as isodata.In maximum likelihood classification, each pixel is tested for all possible classes and the pixel is assigned to the class with the highest posterior probability. With neural networks, once a neural network is trained it directly maps the input observation vector to the output category. Thus for large images neural networks are more suitable. Huang and Lippmann [1] have compared neural networks with conventional classifiers. Eberlein et al. [2] have used neural network models for data analysis by a back-propagation (BP) learning algorithm in a geological classification system. Cleeremans et al. [3] have used neural network models with a BP learning algorithm for Thematic Mapper data analysis which was available on previous versions of Landsat. Decatur [4] has used neural networks for terrain classification. Kulkarni and Lulla[5] have developed software to simulate three models: a three -layer feed forward network with back-propagation learning, a three-layer fuzzy-neural network model, and a four-layer fuzzy-neural network model. The models were used as supervised classifiers to classify pixels based on their spectral signatures. They considered two Landsat scenes. The first scene represents the Mississippi river bottomland area, and the second scene represents the Chernobyl area. Both scenes are of the size 512 by 512 pixels. Each pixel was represented by a vector of five gray values. Bands 2,3,4,5 and 7 were chosen as these bands showed the maximum variance and contained information needed to identify various classes. Kulkarni [6] has analyzed the Mississippi scene also with a self-organizing neural network with a competitive learning algorithm. Clustering algorithms such as the split-merge [7], fuzzy K-means [8], [9], and neural network based methods have been used for multispectral image analysis. Kulkarni and McCaslin[10] have used neural networksfor classification of pixelsin multispectral images and knowledge extraction. Mitra et al. [11] have considered a support vector machine (SVM) for classifying pixels in land use mapping. Decision trees represent another group of classification algorithms. Decision tree classifiers have not been used widely by the remote sensing community despite their non-parametric nature and their attractive properties of simplicity in handling the non-normal, non-homogeneous and noisy data [12].

The Random Forest algorithm has been used in many data mining applications, however, its potential is not fully explored for analyzing remotely sensed images. Random Forest is based on tree classifiers. Random Forest grows many classification trees. To classify a new feature vector, the input vector is classified with each of trees in the forest. Each tree gives a classification, and we say that the tree "votes" for that class. The forest chooses the classification having the most votes over all the trees in the forest. Among many advantages of Random Forest the significant ones are: unexcelled accuracy among current algorithms, efficient implementation on large data sets, andan easily saved structure for future use of pre-generated trees [12]. In this paper, we consider Random Forest Algorithm proposed byBreiman[13]. The outline of the paper is as follows. Section 2 describes decision trees andRandom Forest algorithm. Section 3 provides implementation of Random Forest and examples of classification of pixels in multispectral images.We compare performance of the Random Forest algorithm with other classification algorithms such as the maximum likelihood, support vector machine, and neural network models. Section 4 provides discussion of the findings and concludes.

2. METHODOLOGY

2.1. Decision Tree Classifiers

Decision tree classifiers are more efficient than single-stage classifiers. With a decision tree classifier, decisions are made at multiple levels. Decision tree classifiers are also known as multi-level classifiers. The basic concerns in a decision tree classifier are the separation of groups at each non-terminal node and the choice of features that are most effective in separating the group of classes. In designing a decision tree classifier it is desirable to construct an optimum tree so as to achieve the highest possible classification accuracy with the minimum number of calculations [14]. The binary tree classifier is considered a special case of a decision tree classifier.

Appropriate splitting conditions vary among applications. A node is said to be a terminal node when it contains only one class decision. Three widely used methods include entropy, gini, and twoing. The expected information needed to classify an observation vector in D is given by:

$$entropy(D) = -\sum_{i=1}^{m} p_i \log(p_i) \qquad \textbf{(1)}$$

Where p_i is the non-zero probability than an arbitrary observation vector in D belongs to class C_i [15]. Entropy is a basic measure of amount of information. This is the most widely used splitting condition as it attempts to divide the classes as evenly as possible giving the most information gain between child and parent nodes. Some applications may require that the data be split by the largest homogeneous group possible [16]. For this the gin information gain is used. Gini impurity is the probability that a randomly labelled class, taking into account class distribution and priors, is incorrectly labelled. Information gain using the gini index is defined as:

$$gini(D) = 1 - \sum_{i=1}^{m} p_i^2 \qquad (2)$$

where p_i is the probability that an arbitrary observation vector in D belongs to class C_i[15]. Twoing uses a different strategy to find the best split among cases. It gives strategic splits by, at the top of the tree, grouping together classes that are largely similar in some characteristic. The bottom of the tree identifies individual classes. When twoing, classes are grouped into two super classes containing an as-equal-as-possible number of cases. The best split of the super classes is found and used as the split at the current node. This results in a reduction of class possibilities among cases at each child node and a reduction in impurity.Twoing is defined as:

$$twoing(D) = \frac{p_L p_R}{4}\left[\sum_{i=1}^{m} | p(i \mid D_L) - p(i \mid D_R) |\right]^2 \qquad (3)$$

where L and R refer to the left and right children of a split at node t, $p(i|D)$ is the proportion of cases in D that belong to class i, and P_L and P_R are the proportions of cases that go from D to the left and right children respectively [17].The splitting of data at each node is recursive and continues until a stopping condition is met.

An ideal leaf node is one that contains only records of the same class. In practice reaching this leaf node may require an excessive number of splits which are costly. Splitting too much results in nothing more than a lookup table and will perform poorly for noisy data while splitting too little prevents error in training data from being reduced, increasing the error of the decision tree [18]. The decision to continue splitting can be based on previously mentioned information gain. Less technical methods have also been employed. Node depth, that is length from root to node *t*, is used to identify leaf nodes. Similarly a stopping condition could also be satisfied by thresholding the depth of children of a certain node. Another common method is to threshold the number of existing cases at node *t*. If there are fewer cases than some threshold, splitting does not occur [12].

A variation on the basic decision tree, the ID3 tree, has been found to be not only efficient but extremely accurate for large datasets with many attributes. The idea behind ID3 trees is that given a large training set, only a portion is used to grow a decision tree. The remaining training cases are then put down the tree and classified. Misclassified results are used to grow the tree further and the process repeats. When all remaining cases in the training set are accurately classified the tree is complete. This method will grow an accurate tree much more quickly than growing a tree using the entire training set however it should be noted that this method cannot guarantee convergence on a final tree[19].

In Quinlan's original ID3 representation, entropy was used as a splitting condition and total node purity was used to determine if splitting should continue. The information gain was found by calculating the total amount of information needed for the tree and subtracting the information needed by a tree with a root node *N* being split with attribute *A*. The largest information gain using *A* determined the attribute on which to split at node *N*. The process is recursive. Using this Quinlan was able to build efficient and accurate trees very quickly without using the entirety of large training sets.

2.2.Random Forest

Breiman[20] introduced the idea of bagging which is short for "bootstrap aggregating". The idea is to use multiple versions of a predictor or classifier to make an ultimate decision by taking a plurality vote among the predictors. Twenty five regression trees constructed from bootstrap samples of the training set gave a median error decrease of 40% from a single tree predictor over five datasets. In bagging, it has been proved that as the number of predictors increases, accuracy also increases until a certain point at which it drops off. Finding the optimal number of predictors to generate will yield the highest accuracy. Pal and Mather [21] were able to increase classification accuracy of remotely sensed data by bagging using multiple decision trees.

Random Forest are grown using a collaboration of the bagging and ID3 principles. Each tree in the forest is grown in the following manner. Given a training set, a random subset is sampled (with replacement) and used to construct a tree which resembles the ID3 idea. However, every case in this bootstrap sample is not used to grow the tree. About one third of the bootstrap is left out and considered to be out-of-bag (OOB) data. Also, not every feature is used to construct the tree. A random selection of features is evaluated in each tree. The OOB data is used to get a classification error rate as trees are added to the forest and to measure input variable (feature) importance. After the forest is completed a case can be classified by taking a majority vote among all trees in the forest resembling the bootstrap aggregating idea.

For example, in judicial court a robber may be on trial for stealing. The jury members will classify the robber as guilty or innocent. They have been 'trained' by a random subset of every account of robbery in history. This subset contains all robberies a single juror has seen or heard about. Having been 'trained' by other accounts of robbery, each jury member will take different variables of the trial into consideration. One member might take into account the value of the object stolen, the victim of the robbery, and the robbers age while another jury member might classify the robber's guilt based on gender, his/her religion, and the robber's age. At the end of the trial all jury members vote on the classification of the robber. In this limited example, the jury is the forest, each member is a tree, the robber is the case to be classified, and the pieces of evidence of the trial are the features to be used for classification.

The error rate of the forest is measured by two different values. A quick measurement can be made using the OOB data but, of course, a set of test cases can be put through to forest to get an error rate as well. Given the same test cases, the error rate depends on two calculations: correlation between any two trees in the forest and the strength, or error rate, of each tree. Returning to our jury metaphor, if every member of the jury took only the features of age, gender, and race into account for classification, showing high-correlation between jurors, the jury would come to a correct conclusion about half the time (randomly) as age, gender, and race have almost nothing to do with theft and are only three of the many facts of the trial. The goals are to establish a jury that considers every piece of evidence of the trial and to select jurors who, on their own, are usually right about the final outcome. If jury members are trees, how do we grow trees with low correlation to one another and high in strength? The answer lies in how many variables each tree must consider. If we have M input variables select m of them at random to grow a tree. As m increases correlation and individual tree accuracy also increase and some optimal m will give the lowest error rate. Each tree will be grown by splitting on m variables; m stays constant throughout the forest.

Random Forest can also measure variable importance. This is done using OOB data. Each variable m is randomly permuted and the permuted OOB cases are sent down the tree again. Subtracting the number of correctly classified cases using permuted data from the number of correctly classified cases using non-permuted data gives the importance value of variable m. These values are different for each tree but the average of each value over all trees in the forest gives a raw importance score for each variable [22].We have implemented Random Forest using a software package in R language andanalyzed Landsat images. Implementation and results from our analysis are in the next section.

3. Implementation and Results

For our simulation, we utilized the Random Forest package of the Comprehensive R Archive Network (CRAN) implemented by Liaw and Wiener [23]. We considered two Landsat scenes.

The first scene is of the Mississippi bottomland at 34 19 33.7518 N latitude and 90 45 27.0024 W longitude. The second scene is of Yellowstone National Park at 44345.4761 N latitude and 110 2736.1818 W longitude. Images were acquired by Landsat-8 Operational Land Imager(OLI) on 23 September, 2014 and 18 October, 2014 respectively. The spectral bands for OLI are shown in Table 1[24]. Our tests consider bands 1 through 7.

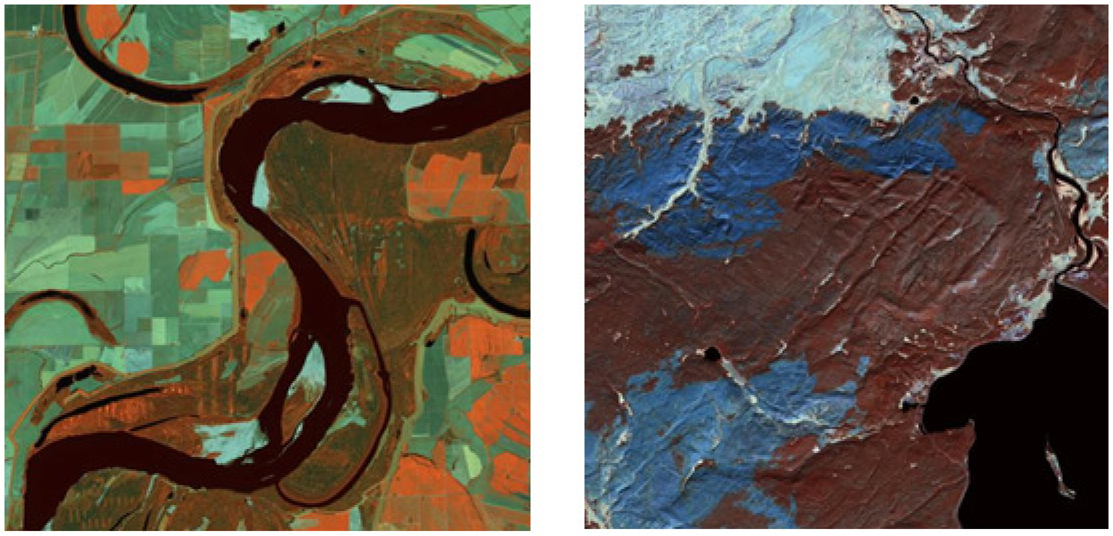

Figure 1. Mississippi and Yellowstone Scenes

Table 1. Landsat Band Descriptions

Bands	**Wavelength (micrometers)**
Band 1 - Coastal aerosol	0.43 - 0.45
Band 2 - Blue	0.45 - 0.51
Band 3 - Green	0.53 - 0.59
Band 4 - Red	0.64 - 0.67
Band 5 - Near Infrared (NIR)	0.85 - 0.88
Band 6 - SWIR 1	1.57 - 1.65
Band 7 - SWIR 2	2.11 - 2.29
Band 8 - Panchromatic	0.5 - 0.68
Band 9 - Cirrus	1.36 - 1.38

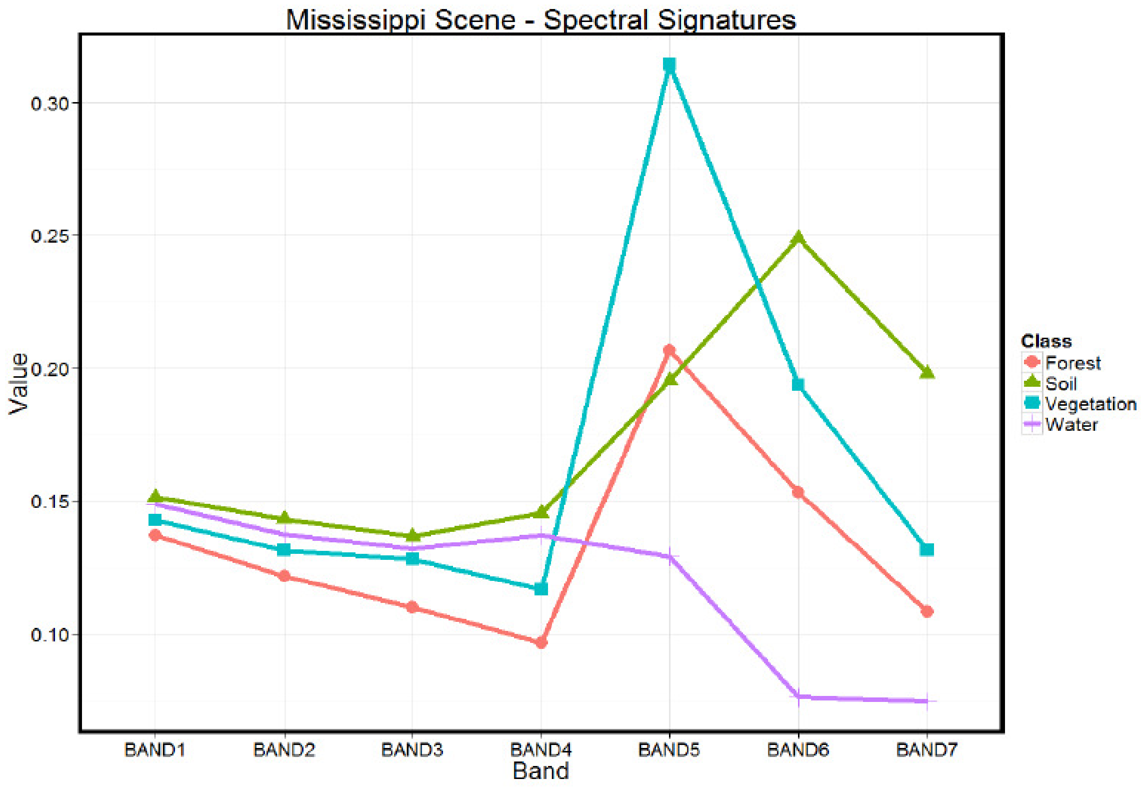

Figure 2. Mississippi Spectral Signatures

We selected subsets of the original scenes of size 512 rows by 512 columns. Both scenes are shown in Figure 1 as a color composite of bands 5, 6 and 7.In order to train each classifier we selected four classes: water, vegetation, soil, and forest. Two training sets for each class, consisting of 100 points each, were selected interactively by displaying the raw image on the computer screen and selecting a 10 x 10 homogeneous area. The classifiers were trained using the training samples and reflectance data for bands 1 through 7.In order to test the classifiers' accuracy, we selected forty test samples and used the spectral signatures as mean vectors for the four classes as shown in Figure 2. The scenes were classified using a random forest, SVM, maximum likelihood classifier, and a neural network. We have assessed the accuracy of the classifiers in the same statistical manner as described by Congalton [25].

Our Random Forest contained 500 trees with an m value of 2. We found that when using a forest this large, the OOB error rate does not significantly change as m grows. The code to implement the Random Forest algorithm using the R package is shown in Figure 3. We used ERDAS Imagine software (version 14) and R Language to implement the other classifiers. Comparing the results of Random Forest with other classifiers yields results found in Table 2 and classified output images are shown in Figure 4.

```
#Import training data from csv
#to train the Random Forest.
>trainingData<-read.csv("200Samples.csv")
>x<-trainingData[,1:7]
>y<-trainingData[,8]

#Create Random Forest with 500 Treesand an m value of 2.
>rf<-randomForest(x,y, mtry=2, ntree=500)

#Import test data, predict values with the RF. Outputconfusioin matrix
>testData<- read.csv("HardBandValues.csv")
>testResults<- predict(rf,testData[,1:7])
>results <- data.frame(correct = testData[,8], prediction=testResults)
>confusionMatrix(results$prediction,results$correct)

#Read pgm files and generate classified image.usingpixmap library

#Function to extract band values for a given point
>bandValues<- function(xc,yc) {
>>c(band1[xc,yc], band2[xc,yc], band3[xc,yc], band4[xc,yc],
band5[xc,yc], + band6[xc,yc], band7[xc,yc])
>>}

#Iterate through entire image and classify each pixel
>final<-array(,c(512,512))

>for (ix in 1:512){for (jy in 1:512) {
>>val<-bandValues(ix,jy)
>>cl<-predict(rf,val)
>>final[ix,jy]<-cl } }

#Write the final output to image file.

#200Samples.csv is a data file containing all band values and expected
#classifications for the chosen 200 samples of each class

#HardBandValues.csv is a data file containing all band values and
#expected classifications for the 40 well-chosen test points of each
#class
```

Figure 3. Random Forest Implementation in R

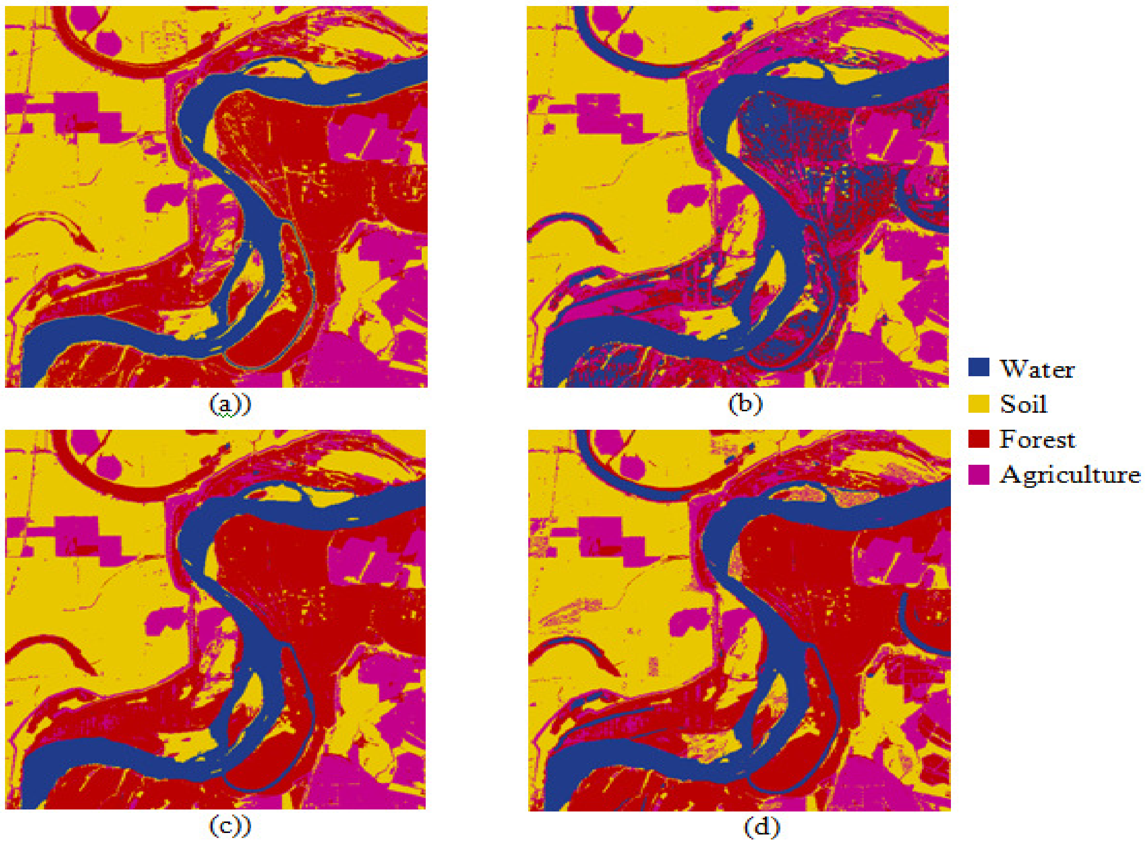

Figure 4. Mississippi Classified Outputs a)Maximum Likelihood, b)Neural Network, c)Support Vector Machine, d)Random Forest

Table 3. Mississippi Accuracy Rates

	Random Forest	**Neural Network**	**Support Vector Machine**	**Maximum Likelihood**
Overall Accuracy	96.25%	76.87%	86.88%	83.13%
Kappa	0.95	0.6917	0.825	0.775
Forest User's Accuracy	88.64%	100%	66.1%	64.91%
Water User's Accuracy	100%	64.51%	100%	100%
Soil User's Accuracy	100%	72.72%	100%	90.91%
Vegetation User's Accuracy	97.22%	100%	97.56%	93.02%
Forest Producer's Accuracy	97.5%	7.5%	97.5%	92.5%
Water Producer's Accuracy	100%	100%	50%	40%
Soil Producer's Accuracy	100%	100%	100%	100%
Vegetation Producer's Accuracy	87.5%	100%	100%	100%

In the Yellowstone scene, we cross-referenced satellite images with forest fire history from the Yellowstone National Park website to give us classes of field, fire damage, forest, and water. The damage from fires Alum, Dewdrop, and Beach, occurring in 2013, 2012, and 2010 respectively, can all be seen in the original image in Figure 1 [26]. It can be seen that over time, the reflectance of a fire damage area will slightly change. We took 200 samples from the Alum fire and 200 samples of the Dewdrop and Beach fires combined as a sample for the fire class. The test was carried out in a similar manner. Figure 5 shows the spectral signatures for the Yellowstone scene. Table 3 shows the accuracy of the various classifiers. The Random Forest grown for the Yellowstone scene contained 500 trees. We found that the OOB error rate was minimized with an m value of 4. Classified output images are shown in Figure 6.

In the Yellowstone scene, we cross-referenced satellite images with forest fire history from the Yellowstone National Park website to give us classes of field, fire damage, forest, and water. The damage from fires Alum, Dewdrop, and Beach, occurring in 2013, 2012, and 2010 respectively, can all be seen in the original image in Figure 1[26]. It can be seen that over time, the reflectance of a fire damage area will slightly change. We took 200 samples from the Alum fire and 200 samples of the Dewdrop and Beach fires combined as a sample for the fire class. The test was carried out in a similar manner. Figure 5 shows the spectral signatures for the Yellowstone scene. Table 3 shows the accuracy of the various classifiers. The Random Forest grown for the Yellowstone scene contained 500 trees. We found that the OOB error rate was minimized with an m value of 4. Classified output images are shown in Figure 6.

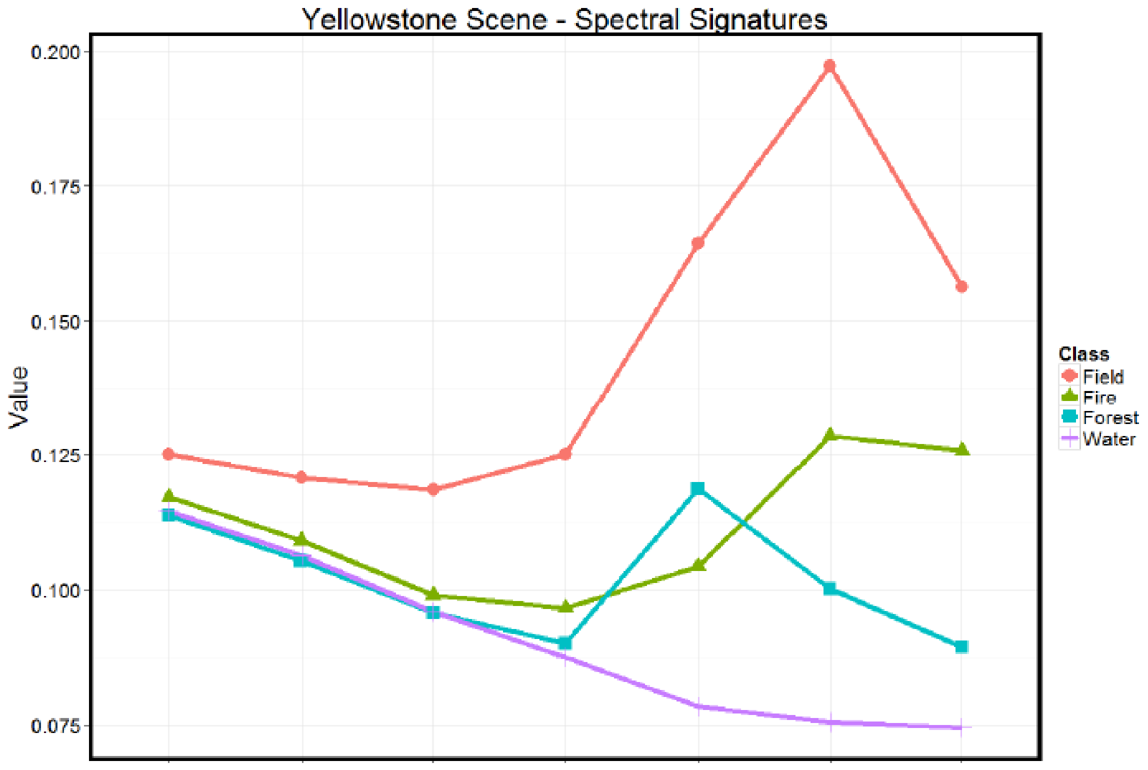

Figure 5. Yellowstone Spectral Signatures

IV. Discussion and Conclusions

In this research we developed simulation for four classifiers: the maximum likelihood, neural network, support vector machine and Random Forest and analyzed two Landsat scenes acquired with Landsat-8 OLI. The scenes were analyzed using ERDAS Imagine and the R package by Liaw and Wiener [23]. It can be seen from Table 2 that the performance of Random Forest was better than all other classifiers in terms of overall accuracy and kappa coefficient. Table 3 shows that Random Forest was outperformed by the neural network and support vector machine. This could be due to impure training sets. Random Forest works well given large homogeneous training data and is relatively robust to outliers

As the Yellowstone scene contained dips in elevation, the reflectance of the bands altered as valleys became shadows. We found that training the forest with the shadowed areas increases the classification error of the forest. Generally, with a large number of training samples, Random Forest performs better [22]. The Mississippi scene was trained with homogeneous samples. This led to high accuracy of Random Forest that outperformed all other classifiers.

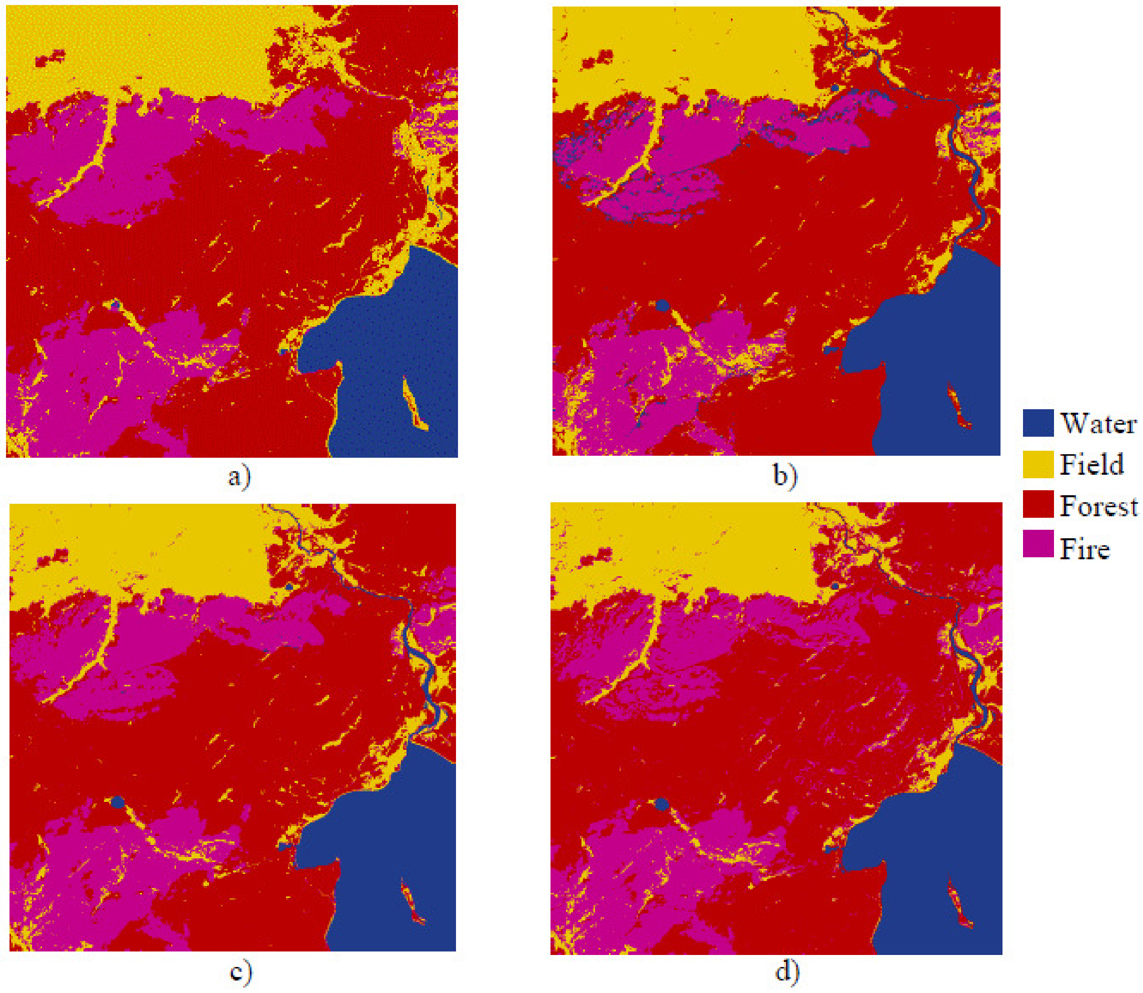

Figure 6. Yellowstone Classified Outputs a) Maximum Likelihood, b)Neural Network, c)Support Vector Machine, d)Random Forest

REFERENCES

[1] W. Y. Huang and R. P. Lippmann, "Neural Net and Traditional Classifiers," in Neural Information Processing Systems, 1988, pp. 387–396.

[2] S. J. Eberlein, G. Yates, and E. Majani, "Hierarchical multisensor analysis for robotic exploration," in SPIE 1388, Mobile Robots V. 578, 1991, pp. 578–586.

[3] A. Cleeremans, D. Servan-Schreiber, and J. L. McClelland, "Finite State Automata and Simple Recurrent Networks," Neural Computation, vol. 1, no. 3, pp. 372–381, Sep. 1989.

[4] S. E. Decatur, "Application of neural networks to terrain classification," in International Joint Conference on Neural Networks, 1989, pp. 283–288 vol.1.

[5] A. D. Kulkarni and K. Lulla, "Fuzzy Neural Network Models for Supervised Classification: Multispectral Image Analysis," Geocarto International, vol. 14, no. 4, pp. 42–51, Dec. 1999.

[6] A. D. Kulkarni, "Neural-Fuzzy Models for Multispectral Image Analysis," Applied Intelligence, vol. 8, no. 2, pp. 173–187, Mar. 1998.

[7] R. H. Laprade, “Split-and-merge segmentation of aerial photographs,” Computer Vision, Graphics, and Image Processing, vol. 44, no. 1, pp. 77–86, Oct. 1988.
[8] R. J. Hathaway and J. C. Bezdek, “Recent convergence results for the fuzzy c-means clustering algorithms,” Journal of Classification, vol. 5, no. 2, pp. 237–247, Sep. 1988.
[9] S. K. Pal, R. K. De, and J. Basak, “Unsupervised feature evaluation: a neuro-fuzzy approach.,” IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council, vol. 11, no. 2, pp. 366–76, Jan. 2000.
[10] A. Kulkarni and S. McCaslin, “Knowledge Discovery From Multispectral Satellite Images,” IEEE Geoscience and Remote Sensing Letters, vol. 1, no. 4, pp. 246–250, Oct. 2004.
[11] P. Mitra, B. Uma Shankar, and S. K. Pal, “Segmentation of multispectral remote sensing images using active support vector machines,” Pattern Recognition Letters, vol. 25, no. 9, pp. 1067–1074, Jul. 2004.
[12] M. Ghose, R. Pradhan, and S. Ghose, “Decision tree classification of remotely sensed satellite data using spectral separability matrix,” International Journal of Advanced Computer Science and Applications, vol. 1, no. 5, pp. 93–101, 2010.
[13] L. Breiman, “Random Forests,” Machine Learning, vol. 45, no. 1, pp. 5–32, Oct. 2001.
[14] A. D. Kulkarni, Computer Vision and Fuzzy Neural Systems. Upper Saddle River, NJ: Prentice Hall, 2001.
[15] J. Han, M. Kamber, and J. Pei, Data Mining: concepts and techniques, 3rd ed. Waltham, MA: Morgan Kaufmann, 2012.
[16] C. Apté and S. Weiss, “Data mining with decision trees and decision rules,” Future Generation Computer Systems, vol. 13, no. 2–3, pp. 197–210, Nov. 1997.
[17] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, Classification and Regression Trees. Belmont, CA: Wadsworth International Group, 1984.
[18] R. O. Duda, P. E. Hart, and D. G. Stork, Pattern Classification, 2nd ed. New York, NY: John Wiley & Sons, Inc., 2001, pp. 394–434.
[19] J. R. Quinlan, “Induction of Decision Trees,” Machine Learning, vol. 1, no. 1, pp. 81–106, Mar. 1986.
[20] L. Breiman, “Bagging predictors,” Machine Learning, vol. 24, no. 2, pp. 123–140, Aug. 1996.
[21] Mahesh Pal and P. M. Mather, “Decision Tree Based Classification of Remotely Sensed Data,” 22nd Asian Conference on Remote Sensing, 2001.
[22] L. Breiman and A. Cutler, “Random Forests,” 2007. [Online]. Available: https://www.stat.berkeley.edu/~breiman/RandomForests/. [Accessed: 08-Aug-2014].
[23] A. Liaw and M. Wiener, “Classification and Regression by randomForest,” R News, vol. 2, no. 3, pp. 18–22, 2002.
[24] “Landsat 8,” 2014. [Online]. Available: http://landsat.usgs.gov/landsat8.php. [Accessed: 11-Nov-2014].
[25] R. G. Congalton, “A review of assessing the accuracy of classifications of remotely sensed data,” Remote Sensing of Environment, vol. 37, no. 1, pp. 35–46, Jul. 1991.
[26] “Wildland Fire Activity in the Park,” 2014. [Online]. Available: http://www.nps.gov/yell/parkmgmt/firemanagement.htm. [Accessed: 10-Nov-2014].

AUTHORS

Barrett Lowe received his bachelor’s degree in drama from the University of North Carolina at Greensboro. He is currently a graduate student in the computer science department at the University of Texas at Tyler. His research interests include data mining, pattern recognition, machine learning, and decision trees. He is a student member of IEEE and aspires to pursue a Ph. D. in computer science.

Dr. Arun Kulkarni, Professor of Computer Science, has been with The University of Texas at Tyler since 1986. His areas of interest include soft computing, data mining, artificial intelligence, computer vision. He has more than seventy refereed papers to his credit, and he has authored two books. His awards include the Office of Naval Research (ONR) 2008 Senior Summer Faculty Fellowship award, 2005-2006 President’s Scholarly Achievement Award, 2001-2002 Chancellor's Council Outstanding Teaching award, and the 1984 Fulbright Fellowship award. He has been listed in who's who in America. He has successfully completed eight research grants during the past twenty years. Dr. Kulkarni obtained his Ph.D. from the Indian Institute of Technology, Bombay, and was a post-doctoral fellow at Virginia Tech.