Available online at www.sciencedirect.com

SCIENCE DIRECT®

Pattern Recognition Letters 27 (2006) 294–300

Pattern Recognition Letters

www.elsevier.com/locate/patrec

Random Forests for land cover classification

Pall Oskar Gislason, Jon Atli Benediktsson *, Johannes R. Sveinsson

Department of Electrical and Computer Engineering, University of Iceland, Hjardarhaga 2-6, IS 107 Reykjavik, Iceland

Available online 5 October 2005

Abstract

Random Forests are considered for classification of multisource remote sensing and geographic data. Various ensemble classification methods have been proposed in recent years. These methods have been proven to improve classification accuracy considerably. The most widely used ensemble methods are boosting and bagging. Boosting is based on sample re-weighting but bagging uses bootstrapping. The Random Forest classifier uses bagging, or bootstrap aggregating, to form an ensemble of classification and regression tree (CART)-like classifiers. In addition, it searches only a random subset of the variables for a split at each CART node, in order to minimize the correlation between the classifiers in the ensemble. This method is not sensitive to noise or overtraining, as the resampling is not based on weighting. Furthermore, it is computationally much lighter than methods based on boosting and somewhat lighter than simple bagging. In the paper, the use of the Random Forest classifier for land cover classification is explored. We compare the accuracy of the Random Forest classifier to other better-known ensemble methods on multisource remote sensing and geographic data.
© 2005 Elsevier B.V. All rights reserved.

Keywords: Random Forests; Classification; Decision trees; Multisource remote sensing data

1. Introduction

During the last two decades multisource remote sensing and geographic data have become increasingly available. As the name indicates, multisource data involves data from multiple data sources and can include, e.g., Landsat satellite multispectral imagery, hyperspectral airborne data, radar data, and geographic data such as elevation and slope (Benediktsson et al., 1990). In land cover classification of remote sensing data, it is desirable to use multisource data in order to extract as much information as possible about the area being classified. However, classification of multisource remote sensing and geographic data is a challenging problem, especially since a convenient multivariate statistical model is in general not available for such data. Consequently, the conventional parametric statistical techniques that have been successfully used in remote sensing data analysis for over four decades are not appropriate (Richards and Jia, 1999) for such classification. Therefore, other methods have been proposed, such as neural network classifiers (Benediktsson et al., 1990) and statistical consensus theory (Benediktsson and Swain, 1992) In the statistical consensus theory approach, heuristic weighting schemes are applied on the data sources involved in the classification. Benediktsson et al. (1997), combined the approaches in (Benediktsson et al., 1990; Benediktsson and Swain, 1992) by optimizing the statistical consensus models with neural networks, and achieved improved classification accuracies over previous studies. However, the hybrid approach in (Benediktsson et al., 1997) has the drawback that it is complicated in implementation because it relies on the parametric statistical modeling of each data source. Therefore, the attention has more recently been turned to the more general approach of ensemble classification.

In ensemble classification, several classifiers are trained and their results combined through a voting process. Many ensemble methods have been proposed (Benediktsson and Swain, 1992; Hansen and Salamon, 1990; Kuncheva, 2003). Perhaps, the most widely used such methods are boosting (Freund and Schapire, 1996; Schapire, 1999)

* Corresponding author. Fax: +354 25 4632.
E-mail addresses: pallgi@hi.is (P.O. Gislason), benedikt@hi.is (J.A. Benediktsson), sveinsso@hi.is (J.R. Sveinsson).

0167-8655/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2005.08.011

and bagging (Breiman, 1994). Bagging (or bootstrap aggregating) is based on training many classifiers on bootstrapped samples from a training set, which has been shown to reduce the variance of the classification. In contrast, boosting uses iterative re-training, where the incorrectly classified samples are given increased weighting as the iterations progress. Consequently, boosting becomes computationally demanding and much slower than bagging. On the other hand, in most cases boosting is considerably more accurate than bagging. Boosting generally reduces both the variance and the bias of the classification and has been shown to be a very accurate classification method. However, it has various drawbacks: it is very slow, it can overtrain and it is sensitive to noise (Briem et al., 2002). Random Forests are ensembles of tree-type classifiers, that use a similar but improved method of bootstrapping as bagging. That is, they can be considered an improved version of bagging. Random Forests have been shown to be comparable to boosting in terms of accuracies, but without the drawbacks of boosting (Breiman, 2001). In addition, the Random Forests are computationally much less intensive than boosting.

Recently, Ham et al. (2005) applied Random Forests to classification of hyperspectral remote sensing data. Their approach is implemented within a multiclassifier system arranged as a binary hierarchy. The obtained experimental results in (Ham et al., 2005) are good for a hyperspectral data set with limited training data. Here, we consider here Random Forests for classification of multisource remote sensing and geographic data (Gislason et al., 2004). As stated above, multisource data are difficult to classify and parametric approaches are not appropriate for classification of such data. The Random Forest approach should be of great interest for multisource classification since the approach is not only nonparametric (Duda et al., 2001) but it also provides a way of estimating the importance of the individual variables (data channels) in the classification.

The paper is organized as follows. In Section 2, the Random Forest classification methodology will be discussed. Experimental results are given in Section 3. Finally, conclusions are drawn in Section 4.

2. Random Forests

Random Forest is a general term for ensemble methods using tree-type classifiers $\{h(\boldsymbol{x}, \Theta_k),\ k = 1, \ldots,\}$ where the $\{\Theta_k\}$ are independent identically distributed random vectors and $\boldsymbol{x}$ is an input pattern (Breiman, 2001; Breiman, http://oz.berkeley.edu/users/breiman/randomforests.html). In training, the Random Forest algorithm creates multiple CART-like trees (Breiman et al., 1984), each trained on a bootstrapped sample of the original training data, and searches only across a randomly selected subset of the input variables to determine a split (for each node). For classification, each tree in the Random Forest casts a unit vote for the most popular class at input $\boldsymbol{x}$. The output of the classifier is determined by a majority vote of the trees.

The number of variables is a user-defined parameter (often said to be the only adjustable parameter in Random Forest), but the algorithm is not sensitive to it. Often, a blindly selected such value is set to the square root of the number of inputs. By limiting the number of variables used for a split, the computational complexity of the algorithm is reduced, and the correlation between trees is also decreased. Finally, the trees in Random Forests are not pruned, further reducing the computational load.

As a result, the Random Forest algorithm can handle high dimensional data and use a large number of trees in the ensemble. This combined with the fact that the random selection of variables for a split seeks to minimize the correlation between the trees in the ensemble, results in error rates that have been compared to those of AdaBoost (Freund and Schapire, 1996) while being computationally much lighter. As each tree is only using a portion of the input variables in a Random Forest, the algorithm is considerably lighter than conventional bagging with a comparable tree-type classifier.

The analysis of Random Forest (Breiman, 2003) shows that its computational time is $cT\sqrt{M}N\log(N)$ where c is a constant, T is the number of trees in the ensemble, M is the number of variables and N is the number of samples in the data set. It should be noted that although Random Forests are not computationally intensive, they require a fair amount of memory as they store an N by T matrix in memory.

Generally, ensemble methods are considered to be black-box type classifiers. Of course there are various methods that can be used to supplement them in order to gain more insight into the data. The nature of ensemble classifiers is such that much information can be garnered by clever use of bi-product data.

To estimate the test set accuracy, the out of bag samples (the remaining training set samples that are not in the bootstrap for a particular tree) of each tree can be run down through the tree (cross-validation). The results are combined with a majority vote as before, and when compared to the actual labels, they yield a lower estimate of the classification error (Breiman, ftp://ftp.stat.berkeley.edu/pub/users/breiman/notes_on_random_forests_v2.pdf), as each sample can only be tested on the trees for which it was out-of-bag (so the effective ensemble for the out-of-bag estimate is smaller than the full ensemble used to train the entire forest).

Similarly, the importance of variable m can be estimated by randomly permuting all the values of the mth variable in the out of bag samples for each classifier (thereby destroying the information contained in that variable). If an increased out-of-bag error is produced, that is an indication of the importance of that variable.

Finally, one can obtain a measure of proximity between two samples, by counting for how many trees they end up in the same terminal node (this is normalized by dividing

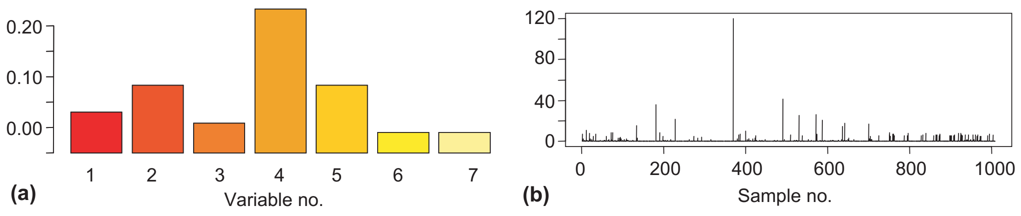

Fig. 1. Colorado training data—(a) variable importance and (b) outlier analysis.

with the number of trees in the ensemble). When this measure is called for, a proximity matrix is computed. The proximity measure can take a significant amount of memory, *NT*, as discussed above. The proximity measure can for example be used to detect outliers. For each sample, the average squared proximity to the other samples in the class is calculated. The raw outlier measure is the number of samples divided by this average proximity (that is, the raw outlier measure will be large if the squared average proximity is small). The median and absolute deviation are calculated for the raw outlier measures of each class, and the raw measure for each sample in that class is then normalized by subtracting the median from the raw score and then dividing by the absolute deviation (Breiman, http://oz.berkeley.edu/users/breiman/randomforests.html). This is the outlier measure shown on Figs. 1 and 3.

3. Experimental results

Classification was performed on a data set consisting of the following four data sources (Benediktsson et al., 1990):

1. Landsat Multispectral Scanner (MSS) data (four spectral data channels).
2. Elevation data (in 10 m contour intervals, one data channel).
3. Slope data (0°–90° in 1° increments, one data channel).
4. Aspect data (1°–180° in 1° increments, one data channel).

The area used for classification is a mountainous area in Colorado. It has 10 ground-cover classes. One class is water; the others are forest types, as specified in Table 1. It is very difficult to distinguish among the forest types using the Landsat MSS data alone since the forest classes show very similar spectral response. The number of available reference samples is 2019 and they are split almost evenly between training and test samples by selecting every other sample for test and the rest for training.

The Random Forest classifier was applied on the Colorado data. A variable importance estimate (as discussed in Section 2) for the training data can be seen in Fig. 1(a) where each data channel is represented by one variable. The first four variables are Landsat MSS data, whereas the latter three are elevation, slope and aspect measurements. It is interesting to note that variable four is the most important variable, followed by variable two and five. That is, elevation is the only topographic variable of importance. The variable importance for each individual class can be seen on Fig. 2. Some interesting conclusions can be drawn from this figure. For example, the classification of water relies mostly on the fourth spectral channel, and elevation variable is important for the classification of all the aspen classes. Also of interest, the slope variable is the second most important variable for the classification of Colorado Blue Spruce.

Table 1
Colorado data—information classes and samples

Class no.	Information class	Training samples	Test samples
1	Water	301	302
2	Douglas Fir/White Fir/Aspen	49	50
3	Colorado Blue Spruce	56	56
4	Mountane/Subalpine Meadow	43	44
5	Aspen	70	70
6	Ponderosa Pine	157	157
7	Ponderosa Pine/Douglas Fir	122	122
8	Engelmann Spruce	147	147
9	Douglas Fir/White Fir	38	38
10	Douglas Fir/Ponderosa Pine/Aspen	25	25
	Total samples	1008	1011

In order to see the changes from the use of a single CART tree to a Random Forest, variable importance analysis was done using individual decision trees based on a leave one out procedure. The results of such analysis for the individual CART are shown in Fig. 3. When Figs. 2 and 3 are compared, the effect of using a forest for classification instead of an individual tree can be seen. In most cases, the variable importance has changed in some way from Fig. 3 to Fig. 2. For example, the individual CART algorithm only uses variable 4 for classification of the Water class whereas the Random Forest uses additional contributions from 5 of the 6 remaining variables. Similar observations can be made regarding the classification of the Ponderosa Pine class (class 2). In contrast, in classification of the Engelmann Spruce class, the CART algorithm uses contributions from all variables where the Random Forest does not use variable 7 and a very limited contribution from variable 6.

A proximity matrix was computed for the training data in order to detect outliers. The results of this outlier

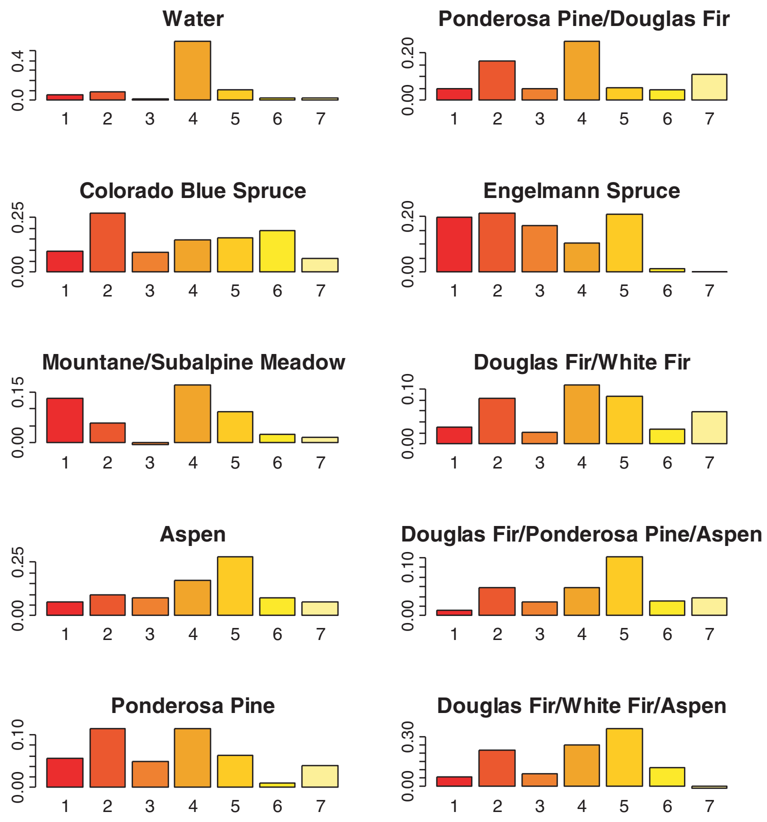

Fig. 2. Colorado training data—variable importance for each of the 10 classes in a Random Forest.

analysis are shown in Fig. 1(b) (all training data) and Fig. 4 (specific classes). Fig. 4 shows that the data set is difficult for classification since there are several outliers. From Fig. 4, the outliers are spread over all classes, however, with a varying degree. It is interesting that outliers are observed even in the water class but those outliers can be attributed to noise in the data.

The experimental results for Random Forest classification are given in Tables 2–4. The overall accuracy (see Table 2) was seen to be insensitive to variable settings, the overall accuracies using the settings based on the square root of the number of inputs (500 trees, and in this case, 2 split variables) were very close to the most accurate result in terms of test accuracy (the most accurate result was obtained with 500 tress and 3 split variables). This is a very important property, as the classifier can be run without human guidance. It is interesting to note that the out-of-bag accuracy estimate is considerably lower than the test set accuracy. As explained previously, this is mostly due to the fact that each bootstrap uses 2/3 of the training samples, so each tree can only use the remaining 1/3 of the training samples to estimate the out-of-bag accuracy. Perhaps a better way to view this is to consider that for the out-of-bag estimate, we can only run each training sample down a third of the entire ensemble of classifiers. That is, the effective ensemble used for the out-of-bag estimate is only a third of the ensemble used to classify the test set, so lower accuracy is to be expected. For a larger training set one would expect the out-of-bag error to approach the test set error.

The training and test accuracies for the individual classes using Random Forests with 500 trees and 3 variables at each node are given in Tables 3 and 4, respectively. From these tables, it is seen that the Random Forests classify water from the forest type classes with 100% accuracy. On the other hand, the algorithm has more difficulty classifying between the individual forest types (classes 2–9) because of the mixed nature of many of the classes.

In the experiments, the Random Forest classifier proved to be extremely fast. Using a Pentium M 1.5 GHz laptop, it took under two seconds to train and classify the data set with the default settings.

The obtained classification accuracies for the Random Forests were compared to results for boosting and bagging with different types of base classifiers (Briem et al., 2002). These base classifiers are:

- Decision table using 10-fold cross-validation for feature selection and termination of search for the best feature set after 15 non-improving subsets (Kohavi, 1995).
- 1R classifier (Witten and Frank, 1993), which is a very simple classifier that operates on only one feature. It splits the data range into buckets (eight in this case). During training, the algorithm looks finds the class from which the most training samples fall into each bin. It then uses this information to classify.

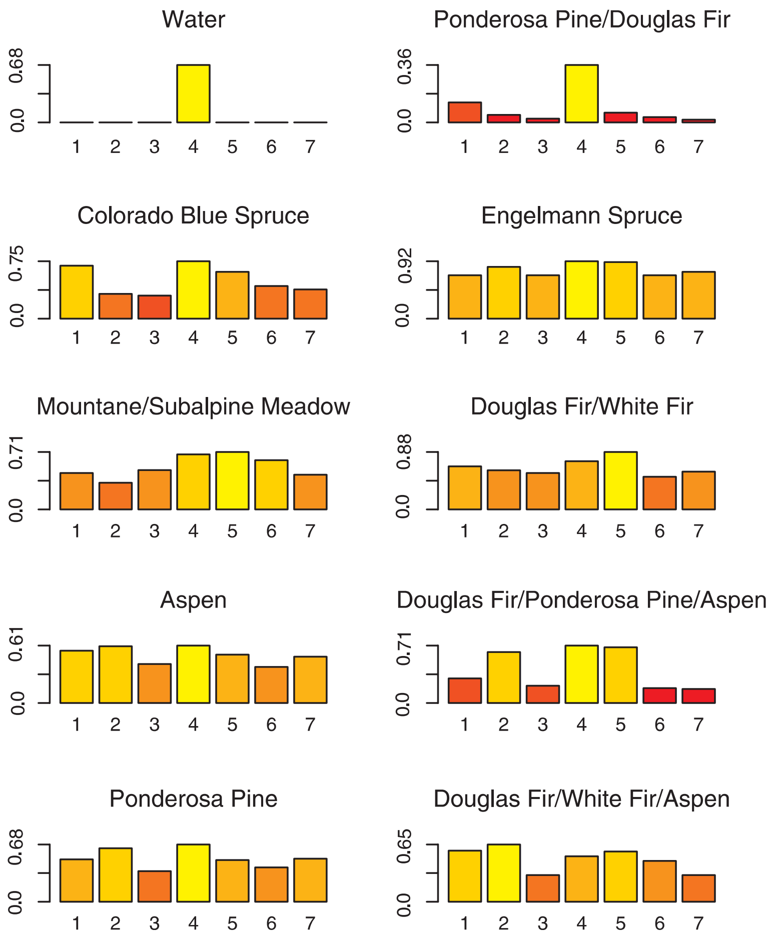

Fig. 3. Colorado training data—variable importance for each of the 10 classes for a single CART tree.

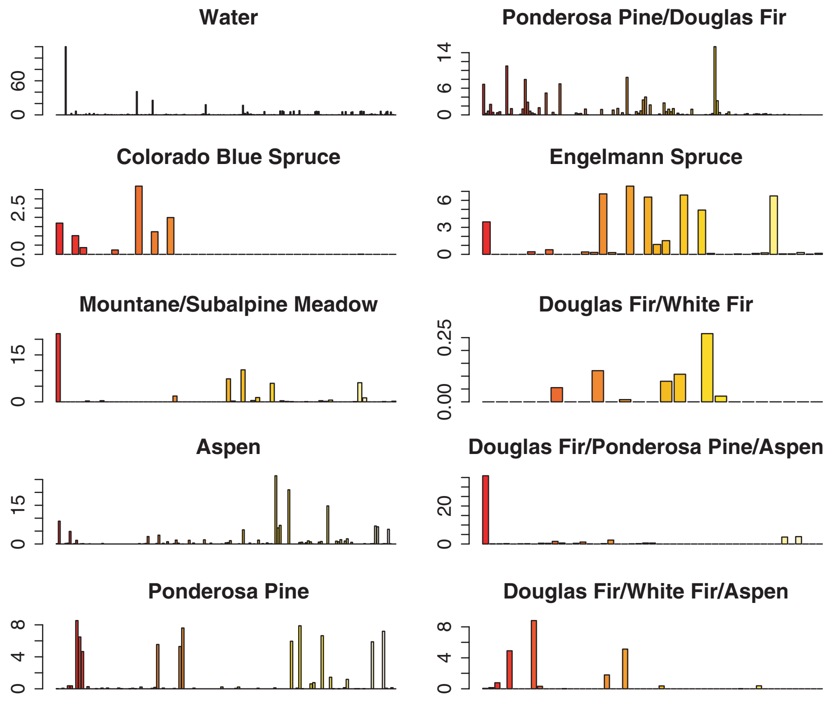

Fig. 4. Colorado training data: outlier analysis for the individual classes. In each case, the x-axis (index) gives the number of a training sample and the y-axis the outlier measure.

Table 2
Colorado data—classification accuracies for the Random Forest classifier

Trees	Split variables	Runtime (min:s)	Out-of-bag acc. (%)	Test set acc. (%)
500	2	00:02	77.47	82.00
1000	2	00:03	79.47	82.10
10,000	2	00:27	78.87	82.39
500	**3**	00:02	77.39	**82.79**
2000	3	00:05	77.68	**82.69**
500	4	00:02	78.47	82.00
10,000	4	00:30	78.47	81.80
2000	4	00:06	78.27	81.80
2000	6	00:07	77.78	80.91
2000	7	00:07	77.48	80.91
10,000	7	00:30	77.38	80.42

- j4.8 decision tree, which is an implementation of the C4.5 revision eight-decision tree (Holte, 1993). With real-valued data, and no missing features (as is the case here), this algorithm is comparable to CART.

A basic CART implementation was also tested for comparison as a non-ensemble tree based classifier. We used the CART algorithm in Matlab's Statistics toolbox.

The results of the classifications are shown in Table 5. From the table it can be seen that the Random Forest outperformed the basic CART algorithm in terms of overall accuracies by more than 4%. These improvements for the Random Forest are results of the differences which were seen when Figs. 2 and 3 were compared. In terms of accuracies, the Random Forest also outperformed bagging based on the 1R algorithm and boosting of the j4.8 decision tree. On the other hand, the accuracies for the Random Forest algorithm, bagging with j4.8, and boosting of a decision table were similar, i.e., they were within a 1% difference of each other. The best overall accuracy was obtained by boosting the 1R base classifier, resulting in more than 2% increase in accuracy as compared to the accuracy obtained by the Random Forest. Although the Random Forest does not in all cases achieve the accuracies of the boosting and bagging, the Random Forest algorithm has the advantage over boosting that it is much faster (boosting is iterative). Furthermore, Random Forest is in general considerably faster than bagging, depending on the base classifier used in bagging.

Table 5
Comparison of Random Forests to boosting and bagging with various base classifiers in terms of overall test set accuracies

Method	Overall accuracy (%)
CART (not ensemble)	78.3
Bagging (Decision table)	82.5
Bagging (j4.8)	81.7
Bagging (1R)	73.6
Boosting (Decision table)	83.8
Boosting (j4.8)	81.5
Boosting (1R)	85.3
Random Forest	82.8

Table 3
Colorado data—confusion matrix for training data in Random Forest classification (using 500 trees and testing 3 variables at each node)

Class no.	1	2	3	4	5	6	7	8	9	10	(%)
1	301	0	0	0	0	0	0	0	0	0	100
2	0	42	0	0	5	0	0	0	0	2	86
3	0	0	41	6	0	9	0	0	0	0	73
4	0	0	13	22	0	8	0	0	0	0	51
5	0	4	0	0	52	7	1	0	1	5	74
6	0	1	16	4	4	90	33	1	7	1	57
7	0	1	2	0	0	44	70	0	5	0	57
8	0	1	0	0	3	0	0	142	0	1	97
9	0	0	4	0	1	10	9	0	13	1	34
10	0	3	0	0	10	1	2	1	1	7	28

Table 4
Colorado data—confusion matrix for test data in Random Forest classification (using 500 trees and testing 3 variables at each node)

Class no.	1	2	3	4	5	6	7	8	9	10	(%)
1	302	0	0	0	0	0	0	0	0	0	100
2	0	41	0	0	5	1	0	1	1	1	82
3	0	0	42	5	0	9	0	0	0	0	75
4	0	0	11	30	0	3	0	0	0	0	68
5	0	4	0	0	53	3	6	0	3	1	76
6	0	4	16	2	2	108	22	0	3	0	69
7	0	1	0	0	0	21	90	0	10	0	74
8	0	0	0	0	1	0	0	146	0	0	99
9	0	2	4	1	4	8	1	0	18	0	47
10	0	2	0	0	15	0	0	1	0	7	28

4. Conclusions

Random Forests were investigated for classification of a multisource remote sensing and geographic data set, which is both a challenging and important classification problem in remote sensing. In experiments, the Random Forest classifier performed well. It outperformed the single CART classifier in terms of accuracies and was comparable to the accuracies obtained by other ensemble methods, i.e., bagging and boosting. However, the Random Forest classifier was much faster in training when compared to the ensemble methods, especially boosting. The Random Forest algorithm does not overfit, and it does not require guidance (although its accuracy can be tweaked slightly by altering the number of variables used for a split). Furthermore, the algorithm can estimate the importance of variables for the classification. Such estimation is of value for feature extraction and/or feature weighting in multisource data classification. The Random Forest algorithm can also detect outliers, which can be very useful when some of the cases may be mislabeled. With this combination of efficiency and accuracy, along with very useful analytical tools, the Random Forest classifier should be considered very desirable for multisource classification of remote sensing and geographic data, where no convenient statistical models are usually available.

Acknowledgments

First, we thank the anonymous reviewers for their constructive comments, which have helped us in greatly improving the paper. This research was supported in part by the Research Fund of the University of Iceland. The Colorado data set was loaned by Roger Hoffer of Colorado State University. The Random Forest implementation software by L. Breiman and A. Cutler was used in experiments. The software is distributed freely on the internet, both as a standalone Fortran program and as a package for the *R* statistics language (http://www.r-project.org). For this paper we used the *R* interface. The assistance of Sveinn R. Joelsson in the preparation of the final version of the manuscript is greatly appreciated.

References

Benediktsson, J.A., Swain, P.H., 1992. Consensus theoretic classification methods. IEEE Trans. Systems, Man Cybernet. 22, 688–704.

Benediktsson, J.A., Swain, P.H., Ersoy, O.K., 1990. Neural network approaches versus statistical methods in classification of multisource remote sensing data. IEEE Trans. Geosci. Remote Sen. 28, 540–552.

Benediktsson, J.A., Sveinsson, J.R., Swain, P.H., 1997. Hybrid consensus theoretic classification. IEEE Trans. Geosci. Remote Sens. 35, 833–843.

Breiman, L., 1994. Bagging predictors. Technical Report No. 421, Department of Statistics, University of California, Berkeley.

Breiman, L., 2001. Random Forests. Mach. Learn. 40, 5–32.

Breiman, L., 2003. RF/tools—A class of two eyed algorithms. In: SIAM Workshop, http://oz.berkeley.edu/users/breiman/siamtalk2003.pdf.

Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J., 1984. Classification and Regression Trees. Wadsworth, Belmont.

Briem, G.J., Benediktsson, J.A., Sveinsson, J.R., 2002. Multiple classifiers applied to multisource remote sensing data. IEEE Trans. Geosci. Remote Sens. 40, 2291–2299.

Duda, R.O., Hart, P.E., Stork, D., 2001. Pattern Classification, second ed. Wiley, New York.

Freund, Y., Schapire, R.E., 1996. Experiments with a new boosting algorithm. In: Machine Learning. Proceedings of the Thirteenth International Conference. pp. 148–156.

Gislason, P.O., Benediktsson, J.A., Sveinsson, J.R., 2004. Random forest classification of multisource remote sensing and geographic data. In: Proceedings 2004 IEEE International Geoscience and Remote Sensing Symposium (IGARSS 2004). pp. 1049–1052.

Ham, J., Chen, Y., Crawford, M.M., Gosh, J., 2005. Investigation of the random forest framework for classification of hyperspectral data. IEEE Trans. Geosci. Remote Sens. 43, 492–501.

Hansen, L.K., Salamon, P., 1990. Neural network ensembles. IEEE Trans. Pattern Anal. Machine Intell. 12, 993–1001.

Holte, R.C., 1993. Very simple classification rules perform well on most commonly used datasets. Mach. Learn. 11, 63–91.

Kohavi, R., 1995. The power of decision tables. In: Proceedings of 8th Euro. Conference on Machine Learning. pp. 174–189.

Kuncheva, L.I., 2003. Fuzzy versus nonfuzzy in combining classifiers designed by Boosting. IEEE Trans. Fuzzy Systems 11, 1214–1219.

Richards, J.A., Jia, X., 1999. Remote Sensing Digital Image Analysis, An Introduction, Third, Revised and Enlarged Edition. Springer, Berlin.

Schapire, R.E., 1999. A brief introduction to boosting. In: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence. pp. 1401–1406.

Witten, I.H., Frank, E., 1993. Data Mining–Practical Machine Learning Tools With Java Implementations. Morgan Kaufmann, San Francisco.