MANY HANDS MAKE LIGHT WORK – ON ENSEMBLE LEARNING TECHNIQUES FOR DATA FUSION IN REMOTE SENSING

Andreas Merentitis and Christian Debes

AGT International
64295 Darmstadt, Germany
E-mail: {amerentitis, cdebes}@agtinternational.com

ABSTRACT

In this paper we discuss the use of ensemble methods in remote sensing. After a review of the relevant state of the art in ensemble learning - inside and outside the remote sensing community - we provide the necessary theoretical background of this research field. This includes a discussion of the bias/variance tradeoff that is a key notion in machine learning and especially ensemble learning. We provide a review of three of the most relevant and prominent techniques in ensemble learning, namely the Random Forest, Extra Trees and the Gradient Boosted Regression Trees algorithms. All algorithms are assessed in terms of their theoretical properties as well as applicability for remote sensing use cases. Finally, in the experimental section we compare their performance in challenging remote sensing datasets with different properties, while discussing again the reasons that the mechanics of each algorithm might give it an advantage under certain conditions.

1. TOPIC MOTIVATION AND SIGNIFICANCE

In recent years ensemble methods have gained significant attention and have been successfully applied in many different fields [1, 2, 3, 4]. Some of their key advantages include inherent support of parallelism, easiness to implement, highly accurate predictions and the ability to handle a very large number of input variables without overfitting. Applications of ensemble learning in areas such as data mining, pattern recognition and predictive analytics showed significant benefit, making it part of every machine learning practitioner's toolkit. It is noteworthy that some of the most famous machine learning competitions such as the Netflix Prize [5, 6], were won using ensemble methods.

As a generically applicable machine learning toolkit ensemble methods also found their way in the remote sensing community. There are several reasons why ensemble methods are highly beneficial in remote sensing applications.

1. Ensemble methods can provide a strong increase in prediction accuracy compared to some baseline methods (e.g., logistic regression that is often used as baseline in several domains of machine learning classification, even though it is less commonly applied for remote sensing applications). At the same time they typically offer similar accuracy (arguably even a bit higher on average across numerous real-world datasets [7]) to other state of the art methods such as SVMs.

2. In many remote sensing applications the training set might not be perfectly representative for the situations in which the classifier is applied later. This holds especially for scenarios with a high dimensionality and large data size. Ensemble methods typically show higher degree of generalisability in such situations.

3. It is often unfeasible to learn one single strong classifier, e.g. a single SVM, if the training dataset is too large. Ensemble methods have better big-O complexity and inherently support parallelism by training a multitude of weak learners, each of them on a small portion of the training set. This allows running the training process on multiple cores or computers which is highly attractive in many situations.

Formally, an ensemble is a technique for combining numerous weak learners in an attempt to produce a strong learner. An ensemble is a supervised learning method, since it has the capacity to be trained and then used to perform predictions. As such, the ensemble also represents a single hypothesis in the solution space. However, this hypothesis is not necessarily contained within the space of the models which were used to construct the ensemble. Therefore ensembles typically have more flexibility in the functions they can represent, which can result in a reduction of model bias [8]. Considering the typical bias-variance decomposition and the bias-variance tradeoff [9], an increase in model complexity is often associated with an increase in variance, since the more complex model is potentially more prone to overfitting the training data. This effect is encountered to a different extent in various ensemble methods, but some of them are specifically designed to reduce the variance part component of the error (e.g., Bagging).

Although numerous variations of tree-based ensembles have been proposed in the literature, three dominant algorithms have emerged from successful application in a variety of application areas. The first of them and perhaps the most well known is the Random Forest (RF) algorithm, as proposed by Breiman [10] which recently was also applied to hyperspectral imagery [11], [9]. The second one is the Extra Trees (ET) algorithm, that pushes the randomization idea further, targeting an even more significant reduction in variance compared to the Random Forest. Finally, the third algorithm, often called Gradient Boosted Regression Trees (GBRT) or Gradient Tree Boosting is of completely different nature, as it aims to reduce mostly model bias. In this paper we are comparing the goals, assumptions and limitations of the three algorithms under a unified framework of bias-variance decomposition. Moreover, we discuss their advantages and disadvantages, both in general as well as in particular for the field of data fusion in remote sensing. We compare their performance in challenging remote sensing datasets with different parameters, and we conclude with practical considerations for the selection of the appropriate algorithm given the data at hand.

It should be noted that the current paper focuses on comparing the tree-based algorithms that are typically available in most machine learning packages (e.g., implementations of these algorithms are freely available in at least Matlab, Python, and R among others) and thus can be easily integrated in a practitioner's toolbox. More recent variants like for example the Rotation Forest that have shown very promising results in the field of remote sensing [12], [13], and [14] but are still not easily available (e.g., there are fewer open source implementations) are not part of the current study. Excluding these recent works, the selected algorithms that we study in the paper can be considered as a generalization of most tree-based ensembles, for example Tree Bagging is a special case of Random Forest (when in every node the full feature set is considered for doing the split [10]) while most of the boosting algorithms can also be derived from Gradient Boosted Regression Trees (e.g., AdaBoost is derived from GBRT using a special cost function, as described in [15]).

The rest of the paper is structured as follows: Section 2 gives an overview of the state of the art in ensemble learning, both inside and outside the field of remote sensing. Section 3 provides the necessary theoretical foundations of ensemble learning while Section 4 provides a detailed summary of the three most widely used ensemble learning techniques including best practices. Experimental results on three remote sensing datasets are shown in Section 5 before the paper ends with a conclusion in Section 6.

2. A SHORT HISTORY OF ENSEMBLE LEARNING

Combining the results of several classifiers in a systematic way has been a significant topic of research since the late 1980s and early 1990s. It was the work by Breiman [16], Clemen [17], Wolpert [8], Hansen and Salamon [18] as well as Freund and Schapire [19] among others that laid the theoretical foundations of the flourishing field of ensemble methods that we see nowadays.

Two-well known methods from this time include Bootstrap Aggregation (Bagging) [16] and Boosting [19]. In Bagging trees are independently constructed using a bootstrap sample of the whole dataset [20, 21]. A majority voting scheme in the end performs the overall decision. In Boosting on the other hand trees are not independently constructed but give information to the following tree constructors yielding a weighted majority vote for the final decision. An adaptive version of Boosting, AdaBoost [22] was developed in 1995, yielding the Gödel prize for their inventors Freund and Schapire in 2003. Still nowadays, AdaBoost is known to be one of the best out-of-the-box classification algorithms and has been applied to numberless machine learning problems.

The next breakthrough in ensemble learning happened when Breiman introduced the Random Forest [10] in 2001. This algorithm combines Breiman's earlier idea of Bagging with randomised feature selection. The Random Forest soon was applied in various domains including biomedicine [23, 24, 25, 26] image classification [27, 28, 29] and various engineering applications [30].

Ensemble methods were applied in several fields within remote sensing, for example in [31], [32], [33], and [34], a trend that is especially prominent in classification of hyperspectral imagery [35, 36, 37, 11, 38, 9, 39]. This is not surprising since hyperspectral image classification can benefit in several ways from ensemble methods. First, it has been shown [11, 38] that prediction accuracy for classification is often increasing when moving from e.g. SVM-based classification appraoches to Random Forests. Second, in this application one is usually faced with having limited training data that might not be perfectly representative to the scenarios it is applied later on. Ensemble methods generally have stronger generalization capabilities which makes them well suited for this case. Finally, especially if the feature representation of the individual classes is high-dimensional training a single strong learner has high computational complexity. Ensemble methods have been shown to significantly decrease training time.

We expect that in the coming years ensemble methods will become the standard tool for selected applications in remote sensing. This holds especially for all classification and data fusion tasks that involve a high-dimensional feature space.

3. THEORY ON ENSEMBLE LEARNING AND THE BIAS VARIANCE TRADEOFF

The bias-variance decomposition distinguishes between (1) the bias error, which is a systematic error component associated with the learning algorithm and the complexity of the hypotheses set, (2) the variance error, which is an error component associated with differences in the selected hypothesis for different training sets and (3) an error component associated with the inherent uncertainty in the domain. However, while this decomposition is easy and intuitive for regression functions and squared error, an one-to-one mapping in the case of multi-class classification problems with different loss functions is not straightforward and various formulations have been proposed [40], [41], [42], [43], and [44]. A generic formulation that offers several desirable theoretical properties was introduced in [43] and we adopt this formulation here.

For a given training set$\{(\mathbf{x}_1, u_o(\mathbf{x_1})), ..., (\mathbf{x}_n, u_o(\mathbf{x_n}))\}$, a learner produces a certain hypothesis f. Given a test point $\mathbf{x_o}$, this hypothesis generates a prediction $f(\mathbf{x_o}) = y$. Assuming again $u(\mathbf{x_o}) = t$ is the true value of the predicted variable for the test point $\mathbf{x_o}$, then a loss function $L(t, y)$ measures the cost of predicting y when the true value is t. Commonly used loss functions are squared loss $L(t, y) = (t - y)^2$, absolute loss $L(t, y) = |t - y|$, and zero-one loss $L(t, y) = 0$ if $y = t$, $L(t, y) = 1$ otherwise. The first two are broadly used in regression while the third one is the default option for classification problems. In the context of a given loss function the goal of learning can be phrased as producing a hypothesis with the smallest possible loss, meaning that the chosen hypothesis minimizes the average $L(t, y)$ over all points, with each point weighted according to its probability.

The optimal prediction y_* for a point $\mathbf{x_o}$ is the prediction that minimizes $\mathrm{E}_t[L(t, y_*)]$, where the subscript t indicates that the expectation is taken with respect to all possible values of t, weighted according to their probabilities given $\mathbf{x}$. The optimal hypothesis is the one for which $f(\mathbf{x})$ = y_* for every $\mathbf{x}$ and even this hypothesis will have non-zero loss. In the case of zero-one loss function, the optimal hypothesis is the Bayes classifier, and its loss is the Bayes rate [45]. Since the same learner generates different models for different training sets, $L(t, y)$ is a function of the training set. This dependency can be alleviated by averaging over training sets. Let **D** be a set of training sets. In this case the quantity of interest is the expected loss $\mathrm{E}_{\mathbf{D},t}[L(t, y)]$, where the expectation is taken with respect to t and the training sets in **D** (i.e., with respect to t and the predictions $y = f(\mathbf{x})$ produced for $\mathbf{x}$ by applying the learner to each training set in **D**). Having this formulation in place, it is possible to define the main prediction for a loss function L and set of training sets **D** as:

$$\mathrm{y}_m^{L,\mathbf{D}} = \arg\min_{y'} \mathrm{E}_{\mathbf{D}}[L(y, y')] \qquad (1)$$

Therefore, the main prediction is the value y' that has the minimum average loss relative to all the predictions. It can be easily derived from this definition that in the case of squared loss function the main prediction is the mean of the predictions, in the case of absolute loss it is the median, and in the case of zero-one loss it is the mode (the prediction with the highest frequency) [43]. Proceeding further in this path, it is possible to define the bias of a learner on a point $\mathbf{x_o}$ as $\mathrm{B}(\mathbf{x_o}) = L(y_*, y_m)$. Therefore, the bias is the loss incurred by the main prediction with respect to the optimal prediction. Similarly, the variance of a learner on a point $\mathbf{x_o}$ can be defined as $\mathrm{V}(\mathbf{x_o}) = \mathrm{E}_{\mathbf{D}}[L(y_m, y)]$. Therefore, the variance is the average loss incurred by predictions with respect to the main prediction. Finally, the noise at point $\mathbf{x_o}$ is $\mathrm{N}(\mathbf{x_o}) = \mathrm{E}_t[L(t, y_*)]$. Therefore, noise is the component of the loss that cannot be avoided, and is incurred independently of the learning algorithm. It is also important to note that bias and variance can be averaged over all points to produce the average bias $\mathrm{E}_{\mathbf{x}}[\mathrm{B}(\mathbf{x})]$ and the average variance $\mathrm{E}_{\mathbf{x}}[\mathrm{V}(\mathbf{x})]$. Building on the previous definitions, it was shown in [43] that, considering a test point $\mathbf{x_o}$ for which the true prediction is t, a learner that predicts y given a training set in **D**, and an arbitrary loss function L, then the following decomposition of $\mathrm{E}_{\mathbf{D},t}[L(t, y)]$ holds:

$$\mathrm{E}_{\mathbf{D},t}[L(t, y)] = c_1\mathrm{E}_t[L(t, y_*)] + [L(y_*, y_m)] + c_2\mathrm{E}_{\mathbf{D}}[L(y_m, y)] \qquad (2)$$

or,

$$\mathrm{E}_{\mathbf{D},t}[L(t, y)] = c_1 N(\mathbf{x}) + B(\mathbf{x}) + c_2 V(\mathbf{x}) \qquad (3)$$

There are several important observations stemming from the analysis of [43]. If we restrict our analysis only on the part that is applicable for classification problems and consequently the zero-one loss function the most important observation is that specifically for zero-one loss, variance can have a subtractive effect and this is derived from a self-consistent definition of bias and variance for zero-one and squared loss, even if the variance itself remains positive [43]. The fact that variance is additive in unbiased examples but subtractive in biased ones has significant implications: if a learner is biased on a given test point, increasing variance can decrease loss.

This behavior is fundamentally different from that of squared loss, but is obtained with the same definitions of bias and variance, purely as a result of the different properties of zero-one loss. In effect, when zero-one loss is the evaluation criterion, there is a much higher tolerance for variance than if the bias-variance decomposition was strictly additive, because the increase in average loss caused by variance on unbiased examples is (partly) offset by its decrease on biased ones. However, on multiclass problems not all variance on biased points contributes to reducing loss. Considering all training sets for which $y \neq y_m$, only some have $y = y_*$, and it is only in these points that loss is reduced. Therefore, the negative effect of variance is exacerbated as the number of classes increases and this can be an important factor in the selection of models or model parameters.

3.1. Bias/variance tradeoff in ensembles

A lot of effort has been focused in explaining formally why ensemble methods work so well. One of the main concepts [16] used to explain why the Bagging ensemble method reduces zero-one loss (typical for classification) was that of an order-correct learner. Let $\mathbf{D}$ be a set of training sets. In this case the quantity of interest is the expected loss $\mathrm{E}_{\mathbf{D},t}[L(t,y)]$, where the expectation is taken with respect to the value of the true function t and the training sets in $\mathbf{D}$ (i.e., with respect to t and the predictions $y = f(\mathbf{x})$ produced for $\mathbf{x}$ by applying the learner to each training set in $\mathbf{D}$). A learner is order-correct on a point $\mathbf{x_o}$ if and only if $\forall_{y \neq y*} P_{\mathbf{D}}(y) < P_{\mathbf{D}}(y_*)$. Breiman [16] showed that Bagging transforms an order-correct learner into a nearly optimal one. We note that order-correctness and bias are closely related: a learner is order-correct on a point $\mathbf{x_o}$ if and only if $\mathrm{B}(\mathbf{x_o}) = 0$ in the case of zero-one loss. The proof can be derived directly from the definitions, considering that y_m for zero-one loss is the most frequent prediction. Schapire et al. [46] proposed an explanation for why the Boosting ensemble method works in terms of the notion of margin. For algorithms like Bagging and Boosting, which generate multiple hypotheses by applying the same learner to multiple training sets, their definition of margin M on a point $\mathbf{x_o}$ for a two class problem can be expressed as follows:

$$M(\mathbf{x_o}) = P_{\mathbf{D}}(y = t) - P_{\mathbf{D}}(y \neq t) \tag{4}$$

for which a positive margin indicates a correct classification by the ensemble, and a negative one an error. More recently [43] it was proven that the notion of margin is closely related to the bias-variance decomposition and specifically, the margin of a learner on a point $\mathbf{x_o}$ can be formulated in terms of its zero-one bias and variance as:

$$M(\mathbf{x_o}) = \pm[2B(\mathbf{x_o}) - 1][2V(\mathbf{x_o}) - 1] \tag{5}$$

with the positive sign applicable if $y_* = t$ and the negative sign applicable otherwise. Therefore the two main formal explanations of why ensemble methods such as Bagging or Boosting work are closely related. Nevertheless, each of the three algorithms that are compared in this paper is making different assumptions and aims at a different bias-variance tradeoff.

4. THREE CATEGORIES OF TREE-BASED ENSEMBLE METHODS

Perhaps the most well known of the tree-based ensemble methods is the Random Forest (RF) algorithm, which was introduced by Breiman [10]. It utilizes both Bagging and random attribute subset selection for achieving diversity between the weak learners. As indicated by various empirical studies [10], [47]), RFs have emerged as serious contesters to state-of-the-art methods such as Boosting [48] and Support Vector Machines [49]. This of course does not mean that Random Forests consistently outperform the previous algorithms, only that they are on average in a similar performance level (arguably even a bit higher [7]) with the same or less computational cost. Some studies have shown that SVM ensembles can even outperform Random Forests, but given the computational complexity of training non-linear SVMs, training an SVM ensemble with a large number of samples per class can have orders of magnitude higher computational cost.

RFs are using as a weak learner the basic Classification and Regression Tree (CART). This weak learner has the property of being very fast to train while also having small bias. It is however unstable as small variations in the training set can result in very different trees. The idea of the RF algorithm is to compensate for this by introducing randomization and then averaging multiple predictors to reduce variance, with a small increase in bias. The randomization in RF is based in two factors: for each tree a bootstrap of the data (typically 2/3 of the full dataset) is used for training. Furthermore, the split in every node is not performed considering the full set of features but only a random subset of them (typically $\sqrt{k}$ for classification and $k/3$ for regression problems, where k is the total number of features). Among these features that are randomly selected for a given node, the one that provides the best split in terms of a selected metric (typically Gini index or entropy) is chosen. The RF trees are never pruned in an attempt to minimize bias. It should be noted that the algorithm falls to the special case of Bagged Trees if the number of features considered in each node is set to the full feature set. Table 1 summarizes the RF algorithm.

The success of RF in achieving significant reduction in variance and improving accuracy has inspired the Extra Trees (ET) algorithm [50] as an effort to push this idea further to even more drastically reduce variance. Similarly to the RF algorithm, it trains unpruned CART trees in a classic top-down method. However, it has two key differences: the nodes are split selecting cut-points completely at random and it uses the entire training set (instead of a bootstrap) to grow each tree. From a bias-variance point of view, the random selection of cut-points coupled with averaging should be able to more strongly reduce variance compared to the RF algorithm. On the other hand, it introduces more bias; this is related to the second difference, where the entire training set is used to grow trees as a way to reduce the increase in bias. It has been proven that the ET algorithm can asymptotically approximate piecewise multi-linear smooth functions as the number of trees is increasing to infinity, while RF remains piecewise constant and non-smooth even as the number of

Table 1. The Random Forest Algorithm

Step 1.	*Setting hyper-parameters*. Set the hyper-parameters of the RF algorithm, e.g., number of trees T and value of the feature set splitting variable $mtry$
Step 2.	*Resampling*. Sample the training data (random sampling with replacement) to create T different subsets of the data, each of size N, with N approximately 2/3 of the complete training set [10].
Step 3.	*Training the decision trees*. • For a given tree node select a subset of predictor variables at random from the set of all the predictor variables. • Use the predictor variable that provides the best split, according to the selected objective function to do a binary split on that node. • At the next node, select a different set of variables at random from all predictor variables and repeat.
Step 4.	*Classifying*. After the training is done and the algorithm operates in classification mode, when a new input is entered it is run down all of the trees. If the range of valid predictions is $\mathcal{C} = \{1, ..., C\}$ where C is the total number of classes, then the estimated probability of predicting class $y \in \mathcal{C}$ for a given point $\mathbf{x_o}$ is: $p(y\|\mathbf{x_o}) = \frac{1}{T}\sum_{t=1}^{T} p_t(y\|\mathbf{x_o})$ with $p_t(y\|\mathbf{x_o})$ being the estimated density of class labels on the leaf of the tth tree [10].

Table 2. The Extra Trees Algorithm

Step 1.	*Setting hyper-parameters*. Set the hyper-parameters of the ET algorithm, e.g., number of trees T and value of the feature set splitting variable $mtry$
Step 2.	*Training the decision trees*. • For a given tree node select a subset of predictor variables at random from the set of all the predictor variables. • For each predictor variable in the subset, if stop split conditions are not met, draw uniformly a random cut-point a_c. • From the set of possible splits that are derived using the random cut-points select the one that maximizes the score of the objective function. • At the next node, select a different set of variables at random from all predictor variables and repeat.
Step 3.	*Classifying*. After the training is done and the algorithm operates in classification mode, when a new input is entered it is run down all of the trees. If the range of valid predictions is $\mathcal{C} = \{1, ..., C\}$ where C is the total number of classes, then the estimated probability of predicting class $y \in \mathcal{C}$ for a given point $\mathbf{x_o}$ is: $p(y\|\mathbf{x_o}) = \frac{1}{T}\sum_{t=1}^{T} p_t(y\|\mathbf{x_o})$ with $p_t(y\|\mathbf{x_o})$ being the estimated density of class labels on the leaf of the tth tree [10].

trees reaches infinity. Table 2 summarizes the ET algorithm.

The third tree ensemble algorithm, Gradient Boosted Regression Trees (GBRT) [15], has a completely different target as it aims to reduce bias. In this algorithm an ensemble of trees is also trained, but these are not unpruned CART trees with small bias that are averaged to reduce variance as in the other methods. Instead these trees have a relatively small, predefined maximum depth (typically 3-8) and high bias. Furthermore, they are not trained in parallel but sequentially, and in every iteration the new tree to be added targets explicitly the samples that are responsible for the current remaining regression error. Due to its iterative nature, the GBRT aglorithm can approximate very complex functions resulting in models with low bias. On the other hand, the algorithm under certain conditions can be fragile to noise and even more to outliers. Table 3 summarizes the GBRT algorithm.

4.1. Effect of hyper-parameters from a bias-variance perspective

Following the formal presentation of the three algorithms, it is important to discuss the number of hyper-parameters that can be tuned and their effect from a bias-variance point of view. This is required in order to better understand the tradeoffs available for each algorithm, as well as to explain their behavior in the experimental section.

Starting from the Random Forest algorithm, the hyper-parameters that mostly affect its performance are the number of features to use for doing node splits and the number

Table 3. The Gradient Boosted Regression Trees Algorithm

Step 1.	*Setting hyper-parameters*. Set the hyper-parameters of the GBRT algorithm, e.g., number of trees T, maximum depth of the trees, and value of the feature set splitting variable $mtry$
Step 2.	*Training the trees*. • Set initial guess to the value of the dependent variable. • 'Upweight' examples that the existing model poorly predicts. • Compute the residuals based on the current model $r_{mi} = y_i - f_{m-1}(x_i)$ where i refers to observations. It should be noted that f_{m-1} refers to the sum of all previous regression trees. • Fit a regression tree (with a fixed depth) to the residuals. • For each terminal node of the tree, compute the average residual. The average value is the estimate for residuals that fall in the corresponding node. • Add the regression tree of the residuals to the current best model f_m.
Step 3.	*Classifying*. After the training is done and the algorithm operates in classification mode, when a new input is entered it is run down the final model f_M (comprised by all the trees) to generate the prediction.

of trees in the ensemble. Regarding the first one, as a heuristic Breiman recommends $\sqrt{k}$ for classification and $k/3$ for regression problems, where k is the total number of features. If sufficient data are available the parameter can be chosen using cross-validation, often resulting in a (typically modest) improvement compared to the heuristic. Generally, a small number favors more aggressive variance reduction at the expense of more bias. Regarding the number of trees, as a bagging variation RF benefits from them without a danger of overfitting, although diminishing returns and computational costs are the limiting factors. Note that in some cases additional options that are not part of the canonical Random Forest are provided, e.g. enabling tree pruning (or setting a maximum tree depth) can result in further variance reduction at the expense of more bias. A similar effect of favoring more variance reduction can be achieved with setting the minimal samples allowed per leaf. Note that if cross-validation is to be used for defining more than one hyper-parameters, then nested cross-validation is required.

The second case is the Extra Trees algorithm and since it is inspired by the RF algorithm it is also characterized by similar hyper-parameters (number of features to consider for each node split, number of trees, maximum tree depth, and minimal samples allowed per leaf). Due to the intrinsic mechanics of the algorithm, setting the parameters to the same values as for RF will typically result in a model that favors less variance at the expense of more bias.

Finally, the GBRT algorithm as a tree-based algorithm is also having similar hyper-parameters, which however often result in different behavior compared to the other algorithms. The first of these hyper-parameters is the number of trees: in contradistinction to RF and ET, a very large number of trees can result to overfitting (depending also on the amount of training data, noise level, etc.) while a very small number will result to underfitting (both of these effects appear due to the iterative nature of the algorithm). The maximum tree depth is also a lot more critical for GBRT, especially for preventing overfitting. Together with the number of trees and the portion of the training set that is received by each tree, these three are typically the hyper-parameters that mostly affect performance and need to be tuned by nested cross-validation. Finally, hyper-parameters like the number of features to consider for each node split and minimal samples allowed per leaf have the same effect as for the previous algorithms, favoring reduction of variance at the expense of bias.

4.2. Algorithm summary and best practices

Following the introduction of the algorithms and the description of their hyper-parameters, we summarize their key properties in Table 4. The goal of this summary is on one hand to capture the basic mechanics of the algorithms in a way that makes any tradeoffs transparent, as well as to guide the decision process based on desired properties of the solution. From the table it is once again clear that the two first algorithms are quite similar in nature, with the Extra Trees algorithm favoring additional variance reduction at the expense of bias compared to a similarly configured Random Forest (i.e., same number of features checked per node, no restrictions on maximum tree depth). Both of the algorithms deal well with noise (typically Extra Trees does excellent in this case) and with outliers (RFs bootstrap-based training per tree offers it an edge here). Furthermore, both algorithms are robust to the hyper-parameter choices, with ET being slightly more stable in some cases. A disadvantage that both algorithms share is the poor interpretability of the resulting model, especially when hundreds of trees are involved.

The third algorithm, as can also be seen from the table, has the least bias and with proper setting of its hyper-parameters can achieve an excellent bias-variance tradeoff for most datasets. However, the model typically depends

	Bias/Variance Tradeoff	Basic Learner	Node Split	Train size per tree	Robustness to noise	Robustness to outliers	Sensitivity to hyper-parameters	Interpretability
Random Forest	Medium/Low V Medium B	CART (parallel)	Best cut on random feature subset	2/3	Medium/High	Medium/High	Medium/Low	Poor
Extra Trees	Lowest V Medium/High B	CART (parallel)	Best random cut on random feature subset	Full	High	Medium	Low	Poor
GBRT	Medium/High V Lowest B	CART (iterative)	Max depth sequential CART (possibly on random feature subset)	Config.	Medium	Medium/Low	Medium/High	Good

Table 4. *Algorithm summary and comparison*

heavily on the values of these hyper-parameters and poor choices can easily result to underfitting or (more commonly) overfitting. Especially maximum tree depth, number of trees, and percentage of the training set used by each tree are very important for controlling the bias/variance tradeoff and should be selected by nested cross-validation. A big advantage of the algorithm is that its iterative additive training results in a model with relatively good interpretability (at least compared to other ensemble models), although interpreting it is still not as trivial as interpreting a single CART or a gradient boosting model that is not using trees as weak learners. The biggest drawback of the GBRT however is its fragility to outliers, especially in cases that the trees see the entire training set. Therefore, it is a good policy to limit the size of the training set used by each tree when the possibility of having outliers in the training set is high.

5. EXPERIMENTAL RESULTS

While providing the necessary theory behind different ensemble learning methods the main focus of the paper is their application to remote sensing use cases. In this paper we provide results for three datasets - the IEEE GRSS 2013 Data Fusion Contest data (University of Houston campus) [11], Indian Pines, and the IEEE GRSS 2014 Data Fusion Contest dataset (Thetford Mines in Qubec).

The three datasets that have been selected for evaluating the algorithms in this version are characterized by completely different properties, so that a large spectrum of cases is assessed. The IEEE GRSS 2013 Data Fusion Contest dataset captures a challenging and varied semi-urban environment with 15 classes. It is well suited for evaluating the ability of the algorithms to use complementary sources of information (hyperspectral and LiDAR) in a classic data fusion setup, as well as to deal with noise on part of the input space (due to the existence of a shadow). For the University of Houston we used the same training set as in the contest. It has been shown in the past that fusing LiDAR and hyperspectral imagery [51, 52, 53] can successfully combine the complementary information of those two data sources including also the use of hierarchical classifiers and ensemble methods [54, 55].

Indian Pines - despite being quite old - is also a challenging dataset of an agricultural area with 16 different classes of crops. Since most of the classes are different types of crops, this dataset is well suited to assess the performance of the algorithms with different feature extraction schemes. For Indian Pines the training set consists of 50 samples for each class that have been randomly chosen from the reference data, except for classes alfalfa, grass/pasture-mowed, and oats that have very few members. Thus, only 15 samples for each of these classes were randomly picked to be used as training samples and all other samples composed the test set.

Finally, the IEEE GRSS 2014 Data Fusion Contest dataset is also capturing a diverse environment and is very interesting from a data fusion perspective as it combines two data sources with a different resolution. Specifically, this dataset involves two modalities, a coarser-resolution longwave infrared (LWIR, thermal infrared) hyperspectral part and a fine-resolution visible (VIS) wavelength range part. The two sources cover an urban area near Thetford Mines in Qubec, Canada, and were acquired and were provided for the Contest by Telops Inc. (Canada). The ground truth includes 7 landcover classes and the mapping is performed at the higher of the two resolutions. For the Thetford Mines we again used the same training set as in the contest.

For each of the two first datasets, we show the overall accuracy of the three algorithms as we progressively add more sophisticated features and domain-level expertise in the problem. Specifically, as a first step we start from the raw spectral bands (plus LiDAR for the University of Houston dataset). Then we consider PCA components, followed by MNF components coupled with some synthetic features (3 vegetation indices and water absorption) [11]. In the fourth step we consider the same setup, adding also a multitude of segmentation maps [9]. Then we perform a feature pre-selection step using RF feature importance [9] before applying each of the algortihms on the selected features with positive influence (MNF+Synth+SelSegm+L). Finally, post processing consisting of Markov Random Field (MRF) segmentation (plus man-made structure correction for the Houston University dataset) is applied on the outcome of classification [11], [38]. The latter is done to have a framework that allows to include object information in the classifier [56].

It should be noted that each of the first five lines of the table fully describes the used features in the respective step, e.g., when transitioning from PCA to MNF we replace the PCA with MNF and do not add them on top of the previous features, unless this is explicitly captured in the description. For example MNF+Synth+SelSegm+L is the same feature set as the previous line but with feature selection for the segmentation maps. The selection of features follows the following reasoning: MNFs have been shown to be superior to PCA as dimensionality reduction method, especially in cases where the data are noisy and have been used successfully in various remote sensing setups [57], [58]. With respect to the feature selection, unsupervised feature selection using Random Forests has shown also promising results for remote sensing (e.g., [9]), while also being a low computational cost method that fits in the tree-based ensemble classification approach we are using in this paper.

For the MRF segmentation we considered the Iterated Conditional Modes algorithm [59] assuming Gaussianity on the class-conditional probability density functions. This approach allows to perform a post processing in which objects of unreasonable size can be filtered out. With respect to the application of post-processing and MRFs, we refer the reader to our previous works in [11], [38] where also the relation of MRF to regularization is discussed.

Houston University	**RF**	**ET**	**GBRT**
Raw+L	75.8	75.0	76.7
PCA+L	85.8	86.8	83.4
MNF+Synth+L	87.9	86.0	85.1
MNF+Synth+Segm+L	80.9	83.0	78.2
MNF+Synth+SelSegm+L	90.6	87.6	85.7
+Post Processing	94.4	93.3	94.6

Table 5. *Algorithm comparison as we progressively add more domain knowledge: Results for Houston University*

From the results provided in Tables 5 and 6 the first thing we can observe is that while transitioning from raw features to PCA and later to MNF brings little benefit for Indian Pines, it has a significant benefit for the University of Houston, where there are a lot of noisy bands, especially under the shadow. With regard to the performance of the algorithms for the two first rows of the tables, we can see that for the University of Houston the performance is comparable, while for the Indian Pines RF performs best, followed by ET, and then GBRT.

Indian Pines	**RF**	**ET**	**GBRT**
Raw	73.4	73.8	68.2
PCA	73.1	71.7	68.9
MNF+Synth	73.8	73.6	69.8
MNF+Synth+Segm	92.9	95.0	94.7
MNF+Synth+SelSegm	94.8	94.9	93.9
+Post Processing	96.3	95.3	95.1

Table 6. *Algorithm comparison as we progressively add more domain knowledge: Results for Indian Pines*

Adding the segmentation maps in (without feature pre-selection) results in significant performance improvement for Indian Pines, while for Houston University the performance degrades across all algorithms. This is probably due to the fact that a lot of the segmentation maps are distorted by the presence of the shadow therefore being poor predictors; interestingly ET is significantly more robust to this effect, due to the random cut-points property that allows randomization to partially average out the skewed predictors.

When we also enable feature pre-selection (using RF for homogeneity) the results improve significantly for Houston University and more moderately also for Indian Pines. In this setup the performance of the three algorithms is similar, with a small advantage for RF in University of Houston. Adding the post-processing step results in very similar performance for RF and GBRT in Houston (followed by ET), while in Indian Pines RF is doing better and the other two algorithms are very close. An interesting effect is that on average GBRT is gaining a lot more from the post-processing step than the other two algorithms, especially for Houston. This is natural if we consider the Markov Random Field segmentation as a regularization post-classification step, where variance is reduced [38]. This is then indeed expected to benefit more the algorithm that favors solutions with smaller bias and more variance, as GBRT is doing.

For the third dataset, because of its increased size, some of the features are very computationaly heavy to derive (e.g., the abundance maps) so for this case we only show the result that uses as features two sets of KMNF components (with and without regularization), the visual bands, and some synthetic features (gradients and segmentation maps). The results for three algorithms, after the classification and the post processing step, are presented in Table 7. It is clear that for this dataset a well-tuned GBRT has the advantage. Note that for the last two rows of the table, for the GBRT algorithm the maximum depth per tree and the portion of train samples that each tree is receiving are selected by nested cross-validation in order to find a good bias/variance tradeoff.

Thetford Mines	**RF**	**ET**	**GBRT**
MNF+Synth+SelSegm (T=100)	84.1	84.8	84.4
+Post Processing (T=100)	89.5	89.0	89.8
MNF+Synth+SelSegm (T=200)	84.4	85.2	86.2
+Post Processing (T=200)	90.0	89.7	91.4
MNF+Synth+SelSegm (T=500)	84.6	85.5	88.1
+Post Processing (T=500)	90.7	90.1	93.9

Table 7. *Algorithm comparison as we progressively add more trees in the ensemble: Results for Thetford Mines*

Houst. Univ.	**RF**	**ET**	**GBRT**
Clas. (50%)	80.0 ± 0.77	82.1 ± 0.58	75.4 ± 1.11
+Post Pr.	82.6 ± 1.05	84.8 ± 0.92	76.6 ± 1.76
Clas. (75%)	79.2 ± 0.81	80.7 ± 0.73	74.1 ± 1.28
+Post Pr.	82.0 ± 2.04	82.4 ± 1.11	77.3 ± 2.87
Clas. (95%)	64.1 ± 0.94	70.3 ± 0.80	65.2 ± 1.57
+Post Pr.	66.2 ± 2.77	72.8 ± 1.46	68.0 ± 3.05

Table 8. *Algorithm comparison as we reduce the size of the training set: Results for Houston University*

5.1. Behavior of the algorithms under difficult conditions

The effect of difficult conditions like outliers, noise, and small training sets in supervised classification has been studied extensively in the broader machine learning domain (although perhaps less so for the case of some ensemble methods). However, in remote sensing many of the works that explicitly deal with noise and outliers are focusing on the problems of spectral unmixing, image reconstruction, and change detection (e.g., [60]) and relatively few works focus on the classification aspect of the problem (e.g., [61], [62]).

In order to evaluate the behavior of the considered ensemble algorithms under challenging conditions, we include results highlighting the effects of the training set size, noise level, and outliers. Specifically, we have conducted three additional experiments. In the first of them, we use the Houston University dataset to estimate the effect of the training set size, as we reduce the number of training samples by 50%, 75%, and 95% (under the constraint that each class has at least 10 training samples). In the second experiment, using again the Houston University dataset, we keep the number of training samples unmodified but we randomly convert 5%, 10%, and 20% per cent of them to a different class, generating outliers (again the constraint is that every class has at least 10 genuine samples). For the two first experiments we use the Houston dataset since it is easy to meet the 10 genuine samples constraint, compared for example to the Indian Pines case (especially if we applied 75% and 95% or undersampling there). The third experiment aims at studying the effect of zero mean Gaussian White Noise and for this case we use Indian Pines since Houston University is already characterized by a significant amount of non-Gaussian noise, due to the presense of the shadow of a cloud. In each of these three experiments, we use as features the MNF components, synthetic features, a selection of segmentation maps, and LiDAR, as described previously (MNF+Synth+SelSegm+L).

In Table 8 we present the classification results for RF, ET, and GBRT with only part of the training data (under the contraint that each class has at least 10 training samples) followed by apllication of MRF segmentation and man-made structure correction.

Since the case of reduced training set size is of particular practical importance for hyperspectral image classification due to the high cost of acquiring labeled samples, it makes sense to study in more detailed what is the bahavior of each algorith in this setting, evaluating visually also the quality of the classification outcomes. Figure 1 shows the final classification outcome using the hyperspectral and LiDAR data that was provided in the 2013 contest with optimized RF (using the full training set) and is provided as reference (RF result from the last line of Table 5). Figures 2, 3, and 4 show that all algorithms face significant problems under the very challenging conditions of getting only 5% of the training data. However, RF is able to keep a reasonable performance in the areas that are not covered by the shadow of the cloud and ET is still able to get correctly some of the big structures (e.g., the highway and some big commercial buildings, etc.) even under the cloud.

Houst. Univ.	**RF**	**ET**	**GBRT**
Clas. (5%)	80.1 ± 0.53	80.7 ± 0.39	77.5 ± 0.57
+Post Pr.	83.7 ± 1.07	85.2 ± 0.94	82.2 ± 1.21
Clas. (10%)	79.4 ± 0.77	78.9 ± 1.02	76.4 ± 1.11
+Post Pr.	81.2 ± 1.80	80.6 ± 1.38	80.3 ± 2.01
Clas. (20%)	78.3 ± 1.46	76.8 ± 1.62	74.0 ± 1.93
+Post Pr.	80.4 ± 2.12	79.0 ± 2.29	77.6 ± 2.58

Table 9. *Algorithm comparison as we add outliers: Results for Houston University*

Ind. Pines	**RF**	**ET**	**GBRT**
Clas. (10^{-5})	87.2 ± 0.46	88.0 ± 0.24	86.8 ± 0.64
+Post Pr.	89.5 ± 0.85	90.5 ± 0.59	92.4 ± 0.77
Clas. ($10^{-4}/2$)	78.0 ± 0.71	78.4 ± 0.47	75.3 ± 1.32
+Post Pr.	81.1 ± 1.27	83.5 ± 1.06	84.0 ± 1.22
Clas. (10^{-4})	71.2 ± 1.38	72.9 ± 1.03	68.6 ± 1.91
+Post Pr.	78.3 ± 2.52	80.4 ± 2.37	80.1 ± 1.64

Table 10. *Algorithm comparison as we progressively add more white Gaussian noise with the specified variance: Results for Indian Pines*

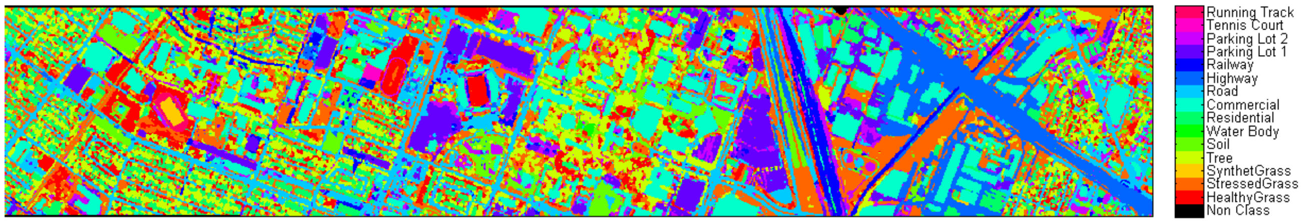

Fig. 1. Classification result for Houston University using RF, segmentation with MRF, and man-made structure correction

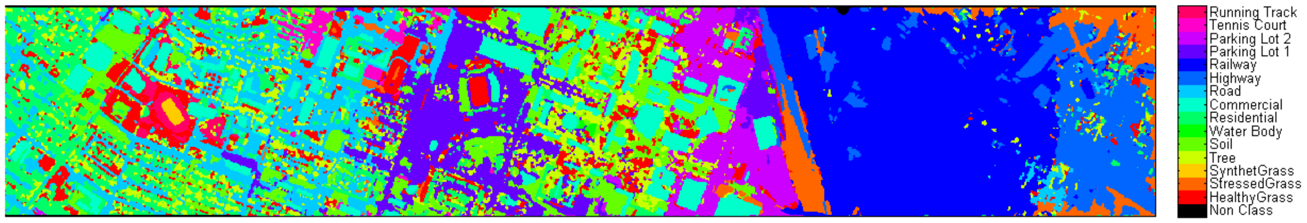

Fig. 2. Classification result for Houston University using 95% downsampled RF, segmentation with MRF, and man-made structure correction

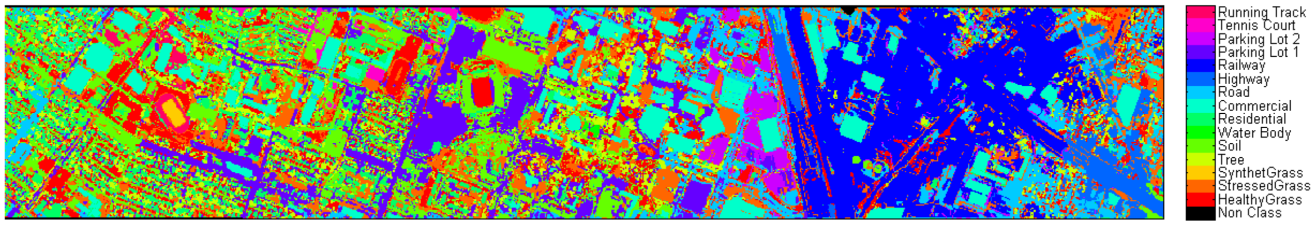

Fig. 3. Classification result for Houston University using 95% downsampled GBRT, segmentation with MRF, and man-made structure correction

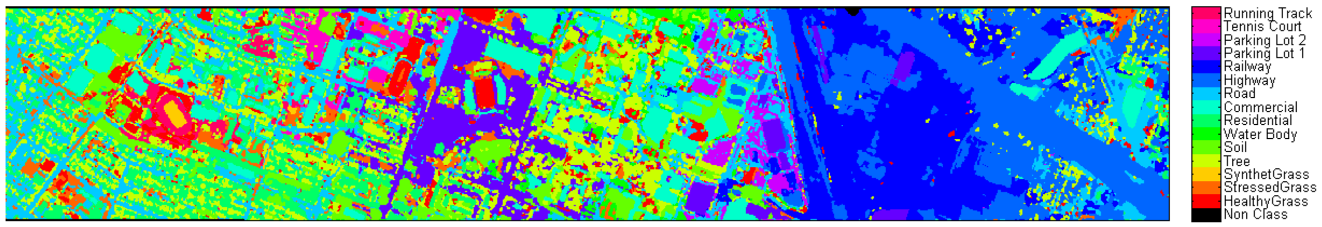

Fig. 4. Classification result for Houston University using 95% downsampled ET, segmentation with MRF, and man-made structure correction

Starting the analysis of the numeric results from the reduced training set experiment, we can further validate the visual impression that the Extra Trees algorithm is doing consistently better compared to the other two and GBRT is the worse on average. Furthermore, while the Random Forest does relatively well with moderate train sample reduction, in the last two rows of the table (reduction of 95%) it is even worse than the GBRT. This is possibly explained by the nature of the algorithms, since ET algorithm is focusing on reducing variance (very important for small training sets) while still using the entire set to train each tree. Apparently, with 95% reduction there are hardly enough training samples available for the RF trees to reach sufficient depth to have low bias and at the same time to exploit randomization in sampling for reducing variance.

For the outlier experiment, GBRT is the least performing algorithm. Between the other two, ET is better when the number of outliers is small, but RF is better when the number of outliers is high. These results are stemming directly from the mechanics of the algorithms, with GBRT being fragile to outliers since it repeatedly tries to find a way to classify them (eventually overfitting them to some extent). For the two other algorithms, the RF uses randomized sampling for each tree in conjunction with randomized feature selection for the node splits, which can result in better control of numerous outliers, compared to the fully randomized cut-points selection of ET, that seems better suited to a smaller number of outliers since it only operates on the features and not on the training samples (therefore the outliers will always appear in all trees).

In the third experiment, where zero-mean White Gaussian Noise with different variances is added to the entire feature space, the performance of the three algorithms is comparable across the different noise levels, with a small advantage for the ET algorithm. It is interesting to note that, while GBRT is typically the least performing of the three when considering only the classification part, it is again typically improving more than the other two algorithms from post-processing, for the reasons that were mentioned previously. It should be noted that this effect is very clear for the particular dataset, since post-processing segmentation is the only step applied here, while for the University of Houston the human structure correction step that is done in addition is favoring algorithms that have a high initial performance for the majority of the pixels in each of the extracted structures and this may balance out the described effect.

Another interesting observation is that in the Gaussian noise setting the good complementarity between GBRT and MRF post-processing can also be observed from the standard deviation of the final result, which is comparable and in some cases less than the respective standard deviation of the final result produced by the other two methods. On the other hand, in the previous setting where outliers were injected, an erroneous classification result from GBRT directly translates to a bad final (post-processed) outcome. Therefore, in the cases that the outliers distorted classification the final outcome was significantly worse compared to cases that the outliers had less impact, thus resulting also in higher standard deviation for GBRT compared to the other two methods.

6. CONCLUSIONS

In this paper we focused on the use of ensemble methods in remote sensing. We briefly described relevant state of the art in ensemble learning, both inside and outside the remote sensing community and elaborated on the theoretical basis of this research field. Following this introduction, we provided a detailed review of three of the most relevant and prominent techniques in ensemble learning, namely the Random Forest, Extra Trees and the Gradient Boosted Regression Trees algorithms. All algorithms were assessed in terms of their theoretical properties as well as applicability for remote sensing use cases. Special care was to taken to describe the effect of hyper-parameters and other configuration choices in a way that is useful for the practitioner in the field of remote sensing. Finally, in the experimental section we compared the performance of the algorithms in challenging remote sensing datasets with different properties. We demonstrated state of the art performance while discussing again the reasons that certain algorithms, due to their mechanics, are in advantage or disadvantage under certain conditions.

7. ACKNOWLEDGEMENT

The authors would like to thank the Hyperspectral Image Analysis group and the NSF Funded Center for Airborne Laser Mapping (NCALM) at the University of Houston for providing the data sets used in this study, and the IEEE GRSS Data Fusion Technical Committee for organizing the 2013 Data Fusion Contest.

8. REFERENCES

[1] R. Polikar, “Ensemble based systems in decision making,” *Circuits and Systems Magazine, IEEE*, vol. 6, no. 3, pp. 21–45, Third 2006.

[2] D. Opitz and R. Maclin, “Popular ensemble methods: An empirical study,” *Journal of Artificial Intelligence Research*, vol. 11, pp. 169–198, 1999.

[3] A. Criminisi, J. Shotton, and E. Konukoglu, “Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning,” *Found. Trends. Comput. Graph. Vis.*, vol. 7, no. 2&#8211;3, pp. 81–227, Feb. 2012.

[4] A. Verikas, A. Gelzinis, and M. Bacauskiene, “Mining data with random forests: A survey and results of new tests,” *Pattern Recognition*, vol. 44, no. 2, pp. 330–349, 2011.

[5] R.M. Bell and Y. Koren, "Lessons from the netflix prize challenge," *ACM SIGKDD Explorations Newsletter*, vol. 9.2, pp. 75–79, 2007.

[6] R.M. Bell, Y. Koren, and C. Volinsky, "All together now: A perspective on the netflix price," *Chance*, vol. 23.1, pp. 24–29, 2010.

[7] Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim, "Do we Need Hundreds of Classifiers to Solve Real World Classification Problems?," *Journal of Machine Learning Research*, vol. 15, pp. 3133–3181, 2014.

[8] D. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5 (2), pp. 241–259, 1992.

[9] A. Merentitis, C. Debes, R. Heremans, and N. Frangiadakis, "Automatic fusion and classification of hyperspectral and lidar data using random forests," in *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium*, 2014.

[10] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[11] C. Debes, A. Merentitis, R. Heremans, J. Hahn, N. Frangiadakis, T. van Kasteren, W. Liao, R. Bellens, A. Pizurica, S. Gautama, W. Philips, S. Prasad, Q. Du, and F. Pacifici, "Hyperspectral and LiDAR data fusion: Outcome of the 2013 GRSS Data Fusion Contest," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, pp. 2405–2418, June 2014.

[12] Taskin Kavzoglu and Ismail Colkesen, "An assessment of the effectiveness of a rotation forest ensemble for land-use and land-cover mapping," *International Journal of Remote Sensing*, vol. 34, no. 12, pp. 4224–4241, 2013.

[13] Junshi Xia, Peijun Du, Xiyan He, and J. Chanussot, "Hyperspectral remote sensing image classification based on rotation forest," *Geoscience and Remote Sensing Letters, IEEE*, vol. 11, no. 1, pp. 239–243, Jan 2014.

[14] A. Samat, P. Du, M.H.A. Baig, S. Chakravarty, and L. Cheng, "Ensemble learning with multiple classifiers and polarimetric features for polarized sar image classification," *Photogrammetric Engineering and Remote Sensing*, vol. 80(3), pp. 239–251, 2014.

[15] Jerome H. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of Statistics*, vol. 29, pp. 1189–1232, 2000.

[16] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, pp. 123–140, 1996.

[17] Robert T Clemen, "Combining forecasts: A review and annotated bibliography," *International journal of forecasting*, vol. 5, no. 4, pp. 559–583, 1989.

[18] L.K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993–1001, 1990.

[19] R.E. Schapire, "The strength of weak learnability," *Machine Learning*, vol. 5, no. 2, pp. 197–227, 1990.

[20] Bradley Efron and B Efron, *The jackknife, the bootstrap and other resampling plans*, vol. 38, SIAM, 1982.

[21] A.M. Zoubir and D. R. Iskander, *Bootstrap Techniques for Signal Processing*, Cambridge University Press, 2004.

[22] Yoav Freund and Robert E Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," in *Computational learning theory*. Springer, 1995, pp. 23–37.

[23] J. Oh, M. Laubach, and A. Luczak, "Estimating neuronal variable importance with random forest," in *Bioengineering Conference, 2003 IEEE 29th Annual, Proceedings of*, March 2003, pp. 33–34.

[24] J. Li, H. Liu, and L. Li, "Diagnostic rules induced by an ensemble method for childhood leukemia," in *Bioinformatics and Bioengineering, 2005. BIBE 2005. Fifth IEEE Symposium on*, Oct 2005, pp. 246–249.

[25] L. Vibha, G.M. Harshavardhan, K. Pranaw, P. Deepa Shenoy, K.R. Venugopal, and L.M. Patnaik, "Statistical classification of mammograms using random forest classifier," in *Intelligent Sensing and Information Processing, 2006. ICISIP 2006. Fourth International Conference on*, Oct 2006, pp. 178–183.

[26] V. Svetnik, A. Liaw, C. Tong, J. Culberson, R. Sheridan, and B. Feuston, "Random forest: A classification and regression tool for compound classification and QSAR modeling," *Journal of Chemical Information and Computer Sciences*, vol. 43, pp. 1947–1958, 2003.

[27] A. Bosch, A. Zisserman, and X. Muoz, "Image classification using random forests and ferns," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, Oct 2007, pp. 1–8.

[28] A.Z. Kouzani, S. Nahavandi, and K. Khoshmanesh, "Face classification by a random forest," in *TENCON 2007 - 2007 IEEE Region 10 Conference*, Oct 2007, pp. 1–4.

[29] S. Bernard, S. Adam, and L. Heutte, "Using random forests for handwritten digit recognition," in *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, Sept 2007, vol. 2, pp. 1043–1047.

[30] W. Yan, "Application of random forest to aircraft engine fault diagnosis," in *Computational Engineering in Systems Applications, IMACS Multiconference on*, Oct 2006, vol. 1, pp. 468–475.

[31] S. Boukir, Li Guo, and N. Chehata, "Classification of remote sensing data using margin-based ensemble methods," in *Image Processing (ICIP), 2013 20th IEEE International Conference on*, Sept 2013, pp. 2602–2606.

[32] Zhongm P. and R. Wang, "A multiple conditional random fields ensemble model for urban area detection in remote sensing optical images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 45, no. 12, pp. 3978–3988, Dec 2007.

[33] Jon Atli Benediktsson, Jocelyn Chanussot, and Mathieu Fauvel, "Multiple classifier systems in remote sensing: From basics to recent developments.," in *MCS*, Michal Haindl, Josef Kittler, and Fabio Roli, Eds. 2007, vol. 4472 of *Lecture Notes in Computer Science*, pp. 501–512, Springer.

[34] M. Pal, "Random forests for land cover classification," in *Geoscience and Remote Sensing Symposium, 2003. IGARSS*

'03. Proceedings. 2003 IEEE International, July 2003, vol. 6, pp. 3510–3512 vol.6.

[35] S.R. Joelsson, J.A. Benediktsson, and J.R. Sveinsson, "Random forest classifiers for hyperspectral data," in *Geoscience and Remote Sensing Symposium, 2005. IGARSS '05. Proceedings. 2005 IEEE International*, July 2005, vol. 1, pp. 4 pp.–.

[36] J. Ham, Yangchi Chen, M. M. Crawford, and J. Ghosh, "Investigation of the random forest framework for classification of hyperspectral data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 3, pp. 492–501, 2005.

[37] P.O. Gislason, J.A. Benediktsson, and J.R. Sveinsson, "Random forest classification of multisource remote sensing and geographic data," in *Geoscience and Remote Sensing Symposium, 2004. IGARSS '04. Proceedings. 2004 IEEE International*, Sept 2004, vol. 2, pp. 1049–1052 vol.2.

[38] A. Merentitis, C. Debes, and R. Heremans, "Application of ensemble learning in hyperspectral image classification: Towards selecting favorable spots in the bias-variance plane," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, pp. 1089 – 1102, April 2014.

[39] J. Chan and D. Paelinckx, "Evaluation of random forest and adaboost tree-based ensemble classification and spectral band selection for ecotope mapping using airborne hyperspectral imagery," *Remote Sensing of the environment*, vol. 112 (6), pp. 2999–3011, 2008.

[40] J. Friedman, "On bias, variance, 0/1-loss, and the curse-of-dimensionality.," *Data Mining and Knowledge Discovery*, vol. 1, pp. 55–77, 1997.

[41] R. Kohavi and D.H. Wolpert, "Bias plus variance decomposition for zero-one loss functions.," in *Proceedings of the thirteenth international conference on Machine Learning*, 1996, pp. 275 – 283.

[42] L. Breiman, "Bias, variance, and arcing classifiers.," *Technical Report 460, Statistics Department, University of California.*, 1996.

[43] P. Domingos, "A unified bias-variance decomposition and its applications.," in *Seventeenth International Conference on Machine Learning*, 2000, pp. 231–238.

[44] G.M. James, "Variance and bias for general loss functions.," *Machine Learning*, vol. 51, pp. 115–135, 2003.

[45] K. Fukunaga, *Introduction to statistical pattern recognition (2nd ed.)*, Academic Press Professional, Inc., San Diego, CA, USA, 1990.

[46] P. Bartlett R.E. Schapire, Y. Freund and W.S. Lee, "Boosting the margin: A new explanation for the efectiveness of voting methods.," in *Proceedings of the Fourteenth International Conference on Machine Learning*, San Francisco, CA, USA, 1997.

[47] J.-M. Poggi R. Genuer and C. Tuleau, "Random forests: Some methodological insights," *arXiv:0811.3619, SSN 0249-6399*, 2008.

[48] Y. Freund and R. Shapire, "Experiments with a new boosting algorithm," in *Proceedings of the 13th International Conference on Machine Learning*, In L. Saitta, Ed., San Francisco, 1996, pp. 14–156.

[49] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*, Cambridge University Press,, 2004.

[50] Pierre Geurts, Damien Ernst, and Louis Wehenkel, "Extremely randomized trees," *Machine Learning*, vol. 63, pp. 3–42, 2006.

[51] M. Dalponte, L. Bruzzone, and D. Gianelle, "Fusion of hyperspectral and LiDAR remote sensing data for classification of complex forest areas," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 46, no. 5, pp. 1416–1427, 2008.

[52] A.F. Elakshe, "Fusion of hyperspectral images and lidar-based dems for coastal mapping," *Optics Lasers Eng.*, vol. 46, pp. 493–498, 2008.

[53] A. Swatantrana, R. Dubayaha, D. Robertsb, M. Hoftona, and J.B. Blairc, "Mapping biomass and stress in the sierra nevada using lidar and hyperspectral data fusion," *Remote Sensing of Environment*, vol. 115, pp. 2917–1930, 2011.

[54] S. Kumar, J. Ghosh, and M. M. Crawford, "Hierarchical fusion of multiple classifiers for hyperspectral data analysis," *Pattern Analysis and Applications*, vol. 5, pp. 210–220, 2002.

[55] Nourzad S and A. Pradhan, "Binary and multi-class classification of fused lidar-imagery data using an ensemble method," *Construction Research Congress*, pp. 909–918, 2012.

[56] G. Moser and S.B. Serpico, "Combining support vector machines and Markov random fields in an integrated framework for contextual image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 51, no. 5, pp. 2734–2752, 2013.

[57] A. A. Nielsen L. Gomez-Chova and G. Camps-Valls, "Explicit signal to noise ratio in reproducing kernel hilbert spaces," in *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium*, 2011, pp. 3570 – 3573.

[58] A. A. Nielsen, "Kernel maximum autocorrelation factor and minimum noise fraction transformations," *IEEE Transactions on Image Processing*, vol. 20 (3), pp. 612–624, 2011.

[59] J. Besag, "On the statistical analysis of dirty pictures," *Journal of the Royal Statistical Society*, vol. 48, pp. 259–302, 1986.

[60] Javier Preciozzi, P. Muse, Andrés Almansa, Sylvain Durand, Francois Cabot, Yann Kerr, A. Khazaal, and B. Rouge, "Sparsity-based restoration of SMOS images in the presence of outliers," in *(IGARSS 2012) IEEE International Geoscience and Remote Sensing Symposium*. July 2012, pp. 3501–3504, IEEE.

[61] Freddy Fierens and Paul L. Rosin, "Filtering remote sensing data in the spatial and feature domains," in *Image and Signal Processing for Remote Sensing, Proc. SPIE*, 1994, pp. 472–482.

[62] Jisoo Ham, Yangchi Chen, Melba M. Crawford, and Joydeep Ghosh, "Investigation of the random forest framework for classification of hyperspectral data.," *IEEE T. Geoscience and Remote Sensing*, vol. 43, no. 3, pp. 492–501, 2005.