Pruning Random Forest with Orthogonal Matching Trees

Luc Giffon, Charly Lamothe, Léo Bouscarrat, Paolo Milanesi, Farah Cherfaoui, Sokol Koço

Pruning Random Forest with Orthogonal Matching Trees

Luc Giffon*[1], Charly Lamothe[1,3], Léo Bouscarrat[1,2], Paolo Milanesi[1], Farah Cherfaoui[1], and Sokol Koço[1]

[1]Aix Marseille Univ, Université de Toulon, CNRS, LIS, Marseille, France
[2]EURA NOVA, Marseille, France
[3]Institute of Neuroscience of la Timone

April 15, 2020

Abstract

In this paper we propose a new method to reduce the size of Breiman's Random Forests. Given a Random Forest and a target size, our algorithm builds a linear combination of trees which minimizes the training error. Selected trees, as well as weights of the linear combination are obtained by mean of the Orthogonal Matching Pursuit algorithm. We test our method on many public benchmark datasets both on regression and binary classification and we compare it to other pruning techniques. Experiments show that our technique performs significantly better or equally good on many datasets [1]. We also discuss the benefit and shortcoming of learning weights for the pruned forest which lead us to propose to use a non-negative constraint on the OMP weights for better empirical results.

Key words: pruning, random forest, OMP.

1 Introduction

Random Forest [Bre01] is an ensemble learning method for supervised classification or regression, which consists in growing a large number of random trees whose output is combined according to some majority vote or average. In the seminal paper on Random Forests [Bre01] it is shown that an increase of the trees number does not cause overfitting but gives instead an asymptotic bound for the generalization error. As a result, the good performance of Random Forest is often obtained at the cost of heavier and less interpretable models. A common approach to overcome these problems is to reduce the size of the Random Forests by mean of pruning techniques [KS12] which denotes, broadly speaking, the process of building a small Random Forest of size K from an initial Forest of size $L \gg K$.

In the rest of this Introduction, we summarize the classification of pruning techniques as proposed in [KS12], then we focus and describe some result pertaining to a subclass of such techniques before introducing our proposed method.

1.1 Classification of forest pruning techniques

The survey [KS12] proposes to classify Forest pruning methods in two different categories called *dynamic* and *static*, which we describe here.

Dynamic pruning. Dynamic pruning techniques conceive pruned forest online: they do not require the preliminary construction of a large set of L trees but instead create the pruned forest of size K from scratch by adding at each step a new tree if it satisfies some criterion which takes into account the current state of the forest [KS12, BAH12]. Although these techniques achieve the final goal of producing a forest of limited size, we esteem that they are not actually *pruning* forest and are out of the scope of this paper.

Static pruning. Static pruning approaches are based on the *Overproduce and choose* paradigm. Techniques from this family start with a large set of L trees and then build the pruned forest of size $K \ll L$ using an iterative process by optimizing a given criterion until the desired size is reached. The techniques are classified as either *forward*, if the final forest is constructed from an empty set by iteratively adding new trees to it, or *backward* if the pruned forest is obtained by removing trees one after the other from the large

*firstname.lastname@lis-lab.fr
[1]Code available at: `https://github.com/swasun/RFOMT`

initial set.

1.2 State of the art in static pruning of random forests

As described in Section 1.1, *static pruning* approaches are iterative algorithms that use an evaluation criterion on trees at each step in order to select the trees that should be either added or removed to the pruned forest. The essential difference between state of the art techniques resides in the choice of the aforementioned criterion. Such criteria essentially build on two salient features that single trees of random forest are expected to incorporate: a high *predictive* performance together with a significant mutual decorrelation, also known as *diversity* in ensemble learning methods [BK10].

In [YLLL12] and [ZW09], the authors remove trees of the forest that have the lowest impact on the margin – *e.g.* the confidence of the forest in its correct classifications – and the accuracy of the forest, respectively. That is, they essentially focus on the prediction performance of trees in the forest. Conversely to [ZW09], in [CNMCK04] the authors propose to make positive selection of trees with replacement, so that they improve the prediction accuracy of the forest at each time step. Finally, the authors of [HYXL07] also perform an accuracy based forward selection before removing those classifiers added after the peak value on validation accuracy. These papers build on the intuition that pruning on the basis of accuracy should give good performance but, as said above, it is known that discrepancy is also a core feature of ensemble models [AP95]. Following this line of thinking, in [ZW09] the authors propose to remove trees with maximum average correlation to other trees in the forest, hence increasing their overall diversity. However, one can think of combining both criteria together –accuracy and diversity – in order to get the best of both worlds. This is the approach by [FME15], where diversity between classifiers is obtained by mean of clustering techniques and [Rok09], where a measure of mutual-information is used to combine the accuracy of trees with the inter-tree mutual agreement for selection.

1.3 Contributions

We propose a static pruning approach with forward search by using the Orthogonal Matching Pursuit (OMP) algorithm [PRK93]. In a nutshell, the novelty of our approach resides in a two-step procedure: first we build a dictionary by using the prediction vectors of the individual trees as atoms; and then we apply OMP in order to select a subset of K atoms (and by extension, trees) whose combination approximate well the true labels of the considered data. Actually, in addition to selecting trees, OMP gives also the coefficient of that linear combinaison. As we will see in Section 3, these coefficients strongly contribute to the quality of our solution. This approach aims at maximizing the prediction accuracy of the forest with the beneficial side effect of encouraging de-correlation between trees, which happens in the core functioning of OMP.

The rest of the paper is organized as follows. In Section 2, we set the notation and we describe how we use the OMP algorithm in the context of Random Forest. In Section 3 we relate the experiments that we conducted with further analysis of the effects of weights on the pruned random forest results. We conclude in Section 4 by discussing the results presented in this paper as well as potential future directions.

2 OMP Forest

2.1 Notations

The Random Forest algorithm trains a collection of L independent decision trees $\{\mathbf{t}_i : \mathcal{X} \rightarrow \mathcal{Y}\}_{i=1}^{L} \subset \mathcal{T}$ to achieve the task of predicting the correct value $\mathbf{y} \in \mathcal{Y}$ from a given $\mathbf{x} \in \mathcal{X}$. We note $\mathbf{t}(\mathbf{x})$ the vector of all tree votes such that $\mathbf{t}(\mathbf{x}) := [\mathbf{t}_1(\mathbf{x}) \dots \mathbf{t}_L(\mathbf{x})]^T$. The Random Forest estimator $\mathbf{f} : \mathcal{T} \times \mathcal{X} \rightarrow \mathcal{Y}$ combines an activation function σ and a linear combination of the votes cast by the underlying decision trees: $\mathbf{f}\left(\{\mathbf{t}\}_{i=1}^{L}, \cdot\right) = \sigma\left(\mathbf{w}^T \mathbf{t}(\cdot)\right)$. The function σ defaults to the identity function for regression ($\mathcal{Y} := \mathbb{R}$) and the *sign* function for binary classification ($\mathcal{Y} := \{-1, +1\}$). The weights are set to $\forall i : \ \mathbf{w}_i := \frac{1}{L}$ as the importance of all trees are usually deemed equal, however arbitrary values can be used.

We can now define a pruned forest decision function in this framework as

$$\mathbf{f}_K\left(\{\mathbf{t}\}_{i=1}^{L}, \cdot\right) := \sigma\left(\mathbf{w}^T \mathbf{t}(\cdot)\right), s.t. \ \|\mathbf{w}\|_0 = K \qquad (1)$$

with K the size of the pruned forest. Thus, the problem of pruning a forest is converted into finding the indices of the optimal K non-zero values in the weight vector $\mathbf{w}$.

2.2 Selecting Trees with OMP

Let $\mathbf{X} \in \mathbb{R}^{N \times d}$ be a matrix of stacked observation vectors and $\mathbf{Y} \in \mathbb{R}^N$ the vector of corresponding true labels. Let $\mathbf{t}_i(\mathbf{X}) := [\mathbf{t}_i(\mathbf{x}_1) \dots \mathbf{t}_L(\mathbf{x}_N)]$ be the predictions vector of the tree $\mathbf{t}_i$ on all the data set $\mathbf{X}$. Thus,

the matrix of the prediction of a set of trees $\{\mathbf{t}_i\}_{i=1}^{L}$ for that dataset can be defined as:

$$\mathbf{T} := [\mathbf{t}_1(\mathbf{X}), \ldots, \mathbf{t}_L(\mathbf{X})]\,. \tag{2}$$

As before, a Random Forest produces its decision based on a linear combination of votes hence the predicted labels associated with $\mathbf{X}$ are $\hat{\mathbf{Y}} = \sigma(\mathbf{T}\mathbf{w})$.

In this work, we propose to achieve the pruning of Random Forests with respect to the training accuracy using the mean square error criterion: $\|\hat{\mathbf{Y}} - \mathbf{Y}\|^2$. Without loss of generality, we drop the σ function in (1) as to simplify the problem into finding:

$$\underset{\mathbf{w}\in\mathbb{R}^L}{\arg\min} \ \|\mathbf{T}\mathbf{w} - \mathbf{Y}\|^2, \text{s.t. } \|\mathbf{w}\|_0 \leq K \tag{3}$$

where the ℓ_0 constraint achieves the pruning objective, *i.e.* the K elements of $\mathbf{w}$ with non-zero values indicate which trees to keep in the pruned forest. As the problem of Equation (3) is known to be NP-hard [DMA97] we propose to use the OMP algorithm [PRK93] which is one of the most adapted algorithm for this task.

Despite its difficulty, the problem of Equation (3) has attracted a lot of attention and is at the center of many signal processing research problems. Indeed, just a few years after Davis Geoff and his co-authors introduced this problem in [DMA97], many different techniques were proposed to give it an approximate solution. Among them, two greedy algorithms have stood out: the Matching Pursuit [DMA97] and its improved counterpart, the Orthogonal Matching Pursuit [PRK93].These two algorithms became famous for their very simple and intuitive form but also for their strong theoretical guarantees [Tro04, GV05]. In this work, we focus our attention on the OMP algorithm which has proven better than MP in many case and which has this pleasant orthogonality property, as we will see later how we benefit from.

In the following, we use the signal processing terminology and call $\mathbf{T}$ the *dictionnary* and its columns the *atoms*; We note that the atoms must be normalized for the OMP algorithm to work.

Orthogonal Matching Pursuit is an iterative algorithm that selects a new atom at each time step to refine its solution. For a given time-step $\tau \geq 0$, we denote by $\mathbf{w}_\tau$ the current solution and by $\mathbf{r}_\tau$ the residual:

$$\mathbf{r}_\tau = \mathbf{Y} - \mathbf{T}\mathbf{w}_\tau. \tag{4}$$

OMP stores the atom which has the largest inner product with the residual $\mathbf{r}_\tau$ and then project the target vector $\mathbf{y}$ onto the span of selected atoms These steps are repeated until the desired number of atom is obtained. The overall procedure is summarized in Algorithm 1.

Roughly speaking, the OMP algorithm increases by one the set of non-zero indexes of $\mathbf{w}$ at each time step and then re-compute the best coefficients of the linear combination. We remark here that the residual is always orthogonal to all selected atoms and then the next chosen atom should be somewhat a different atom from the previously selected ones. In the context of forest pruning, this last claim provides further justification as to why OMP is beneficial for this task: not only the algorithm focus on finding a linear combination of trees that approximate well the true labels, it also encourages diversity between trees.

Algorithm 1 Orthogonal Matching Pursuit [PRK93]

Input: $K, \mathbf{Y}, \mathbf{T} := [\mathbf{t}_1(\mathbf{X}), \ldots, \mathbf{t}_L(\mathbf{X})]$
Output: $\mathbf{w}$, an approximate solution of Problem (3)

$\mathbf{r}_0, \mathbf{w}_0, \lambda = \mathbf{Y}, 0, \emptyset$
for $\tau = 0, .., K$ **do**
  $i^* \in \underset{i\in\{1,\ldots,L\}}{\text{argmax}} \ |\langle \mathbf{t}_i(\mathbf{X}), \mathbf{r}_\tau \rangle|$
  $\lambda = \lambda \cup \{(\mathbf{t}_i(\mathbf{X}))\}$
  $\mathbf{w}_{\tau+1} \in \underset{\substack{\mathbf{w}\in\mathbb{R}^L \text{ s.t.} \\ \mathbf{T}\mathbf{w} \in \text{span}(\lambda)}}{\text{argmin}} \ ||\mathbf{Y} - \mathbf{T}\mathbf{w}||_2^2$
  $\mathbf{r}_{\tau+1} = \mathbf{Y} - \mathbf{T}\mathbf{w}_{\tau+1}$
end for
return $\mathbf{w}_{\tau+1}$

2.3 OMP and Random Forests from the Gradient Boosting perspective

As previously mentioned, the key intuition on applying OMP to forest pruning is that the resulting forest should contain good enough classifiers while providing some diversity among the selected trees. This procedure enjoys good theoretical error bounds on the training set due to the greedy nature of OMP and the choice of the minimization objective in Eq 3. Alas, results on the generalization error are not as straightforward.

In this section we argue that under certain conditions which are to be defined, applying OMP to an ensemble of random trees results in a (gradient) boosting procedure, and this equivalence allows our proposed method to inherit generalization error bounds from recent works on gradient boosting.

To the best of our knowledge, the link between OMP and ensemble methods has yet to be fully studied, however several authors have pointed out the connection between Boosting and Matching pursuit [LRK+18], [Fri01]. In a nutshell, [Fri01] cast boosting as a coordinate descent approach in the function space. The goal of the method is then to find at each iteration

the direction providing the steepest descent. This is equivalent to applying Matching Pursuit on a carefully chosen dictionary and signal.

More formally, let $\mathbf{T} := \{\mathbf{t}|\mathbf{t} : \mathcal{X} \to \mathcal{Y}\}$ be a set of weak classifiers (we voluntarily abuse notation and use $\mathbf{T}$ to define a matrix as in (2) and a set of function here).Thus, the goal of boosting algorithm is to find $\mathbf{f}^* \in lin(\mathbf{T}) := \{\mathbf{t}|\mathbf{t}(X) = \sum_{\mathbf{s}\in\mathbf{T}} \alpha_{\mathbf{s}}\mathbf{s}(X),\ \alpha_{\mathbf{s}} \in \mathbb{R}\}$ such that:

$$\mathbf{f}^* = \arg\min_{\mathbf{f}\in lin(\mathbf{T})} \ell(\mathbf{Y}, \mathbf{f}(X)),$$

where $\ell : \mathcal{Y}^N \times \mathcal{Y}^N \to \mathbb{R}$ is an arbitrary loss function. In the rest of this section, we consider the case where ℓ is the least squared loss function. The boosting algorithm builds the solution $\mathbf{f}_k$ in a greedy fashion, by seeking the classifier which correlates the most with the steepest descent direction, given by

$$-\nabla_{\mathbf{f}(X)}\ell(\mathbf{Y}, \mathbf{f}(X))|_{\mathbf{f}(X)=\mathbf{f}_{k-1}(X)} = \mathbf{r}_k \tag{5}$$

where the definition of $\mathbf{r}_k$ is given in (4). Thus, at each iteration k, the two following steps are repeated to select the new weak classifier and the coefficient, and to update the solution:

$$\mathbf{t}^*, \alpha^* = \arg\min_{\mathbf{t}\in\mathbf{T};\alpha\in\mathbb{R}} \ell(\mathbf{r}_k, \alpha\mathbf{t}(X)), \tag{6}$$

$$\mathbf{f}_k(X) = \mathbf{f}_{k-1}(X) + \alpha^*\mathbf{t}^*(X).$$

By carrying out the computation of (6), one gets

$$\begin{aligned}
\mathbf{t}^* &= \arg\min_{t\in\mathbf{T}} \frac{1}{2}||\mathbf{r}_k - \alpha\mathbf{t}(X)||_2^2 \\
&\overset{(a)}{=} \arg\min_{\mathbf{t}\in\mathbf{T}} \frac{1}{2}||\mathbf{r}_k||_2^2 - \langle\mathbf{r}_k, \alpha\mathbf{t}(X)\rangle + \frac{1}{2}\alpha^2 \\
&= \arg\max_{\mathbf{t}\in\mathbf{T}} |\langle\mathbf{r}_k, \mathbf{t}(X)\rangle| \\
\alpha^* &= \arg\min_{\alpha\in\mathcal{R}} -\alpha\langle\mathbf{r}_k, \mathbf{t}(X)\rangle + \frac{1}{2}\alpha^2 = \langle\mathbf{r}_k, \mathbf{t}(X)\rangle
\end{aligned}$$

where (a) holds because the atoms are normalized. Besides, as we are looking for the classifier pointing at the steepest descent direction, the scalar product $\langle\mathbf{r}_k, \mathbf{t}(X)\rangle$ is essentially positive [MBBF00].

This equivalence is extremely interesting for our pruning approach. By setting the function space $\mathbf{T}$ as the ensemble of all the base trees trained by the random forest and the loss as the squared-error one, then applying MP on this setting results in performing a boosting procedure.

In the general case, this equivalence does not carry over from Matching Pursuit to Orthogonal Matching Pursuit, however there is one particular case when this equivalence holds. Indeed, if Assumption 1 holds, then [Tro04, GV05] proved that OMP and MP converge to the same solution, which is the optimal one. Consequently, OMP is also equivalent to Gradient Boosting.

Assumption 1. *For a given dictionary* $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_L] \in \mathbb{R}^{N\times L}$*, and a target vector* $\mathbf{Y}$*, there exists a vector* $\mathbf{w}$ *such that:* $||\mathbf{w}||_0 = K \leq L$*;* $\mathbf{Y} = \mathbf{T}\mathbf{w}$*; and* $K \leq \frac{1}{2}\left(\frac{1}{\max_{i\neq j}|\langle\mathbf{t}_i,\mathbf{t}_j\rangle|} + 1\right)$.

In the context of forest pruning, this assumption claims that there exists a subset of K forest that can exactly recover the true label for every example in the training set. Since Random Forests produce an important number of random trees, Assumption 1 doesn't seem to be unreasonable.

This boosting-like nature of forest pruning with OMP allows us to use theoretical guarantees on the generalization error of the resulting classifier. In [CMS19], the authors propose a general learning setting for gradient boosting. In particular they study the case where a regularization term on the weak classifier is injected at each step of the gradient boosting procedure. In other terms, the choice of the descent direction is limited to a subspace of all the possible directions. This is exactly the setting of the boosting-like procedure we built with OMP and Random Forests. The authors gave a bound on the generalization error of the constructed classifier. Hence, since this constructed classifier is the same classifier constructed by OMP, the generalization error bound provided in Theorem 2 of [CMS19] is also applicable to OMP, and by extension to the method proposed in this paper.

3 Experiments

In this Section, we start with describing the experimental setting then we describe our results on many datasets with a particular attention on the impact of weights on the performance.

3.1 Experimental setting

Datasets. Experiments have been conducted on 11 datasets for regression or binary classification. A summary of the used datasets characteristics is available in Table 1 along with the size of the base forest for each dataset.

Competing methods. We compare our technique to multiple state of the art pruning forest methods. Following the classification presented in Section 1.2

Dataset name	Acronym	Size	Dimension	Task (score metric)	Base Forest size
Boston house prices [BKW80]	Bos.	506	13	Regression (MSE)	100
Breast cancer [MSW95]	B.C.	569	30	Binary classification (Accuracy)	1000
California housing [PB97]	C.H.	20 640	8	Regression (MSE)	1000
Diabetes [EHJ+04]	Diab.	442	10	Regression (MSE)	108
Diamonds [Dia]	Diam.	53 940	9	Regression	429
Gamma [BCG+04]	Gam.	19 020	11	Binary classification (Accuracy)	100
Kin8nm [Cor96]	Kin.	8 192	8	Regression (MSE)	1000
King-Rook vs. King-Pawn [Sha87]	KR-KP	3 196	36	Binary classification (Accuracy)	1000
LFW pairs [HMBLM08]	LFW	13 233	5 828	Binary classification (Accuracy)	1000
Spambase [CL98]	Sp. B.	4 601	57	Binary classification (Accuracy)	1000
Steel Plates [Bus98]	St. P.	1941	33	Binary classification (Accuracy)	1000

Table 1: Datasets characteristics. Experiments used a 60-20-20% split for train-validation-test.

and Section 1.1, we use *Ensemble* [CNMCK04] as a forward, accuracy based pruning; *Zhang Predictions* [WZ09] as backward, accuracy based pruning; *Zhang Similarity* [WZ09] as backward, diversity based pruning; and finally *Kmeans* [FGE15] which is forward and both an accuracy and diversity based approach. We also compare our results to a *Random* selection technique which is equivalent to training directly a forest of the desired size.

Implementation details. All the algorithms were implemented in Python, using the scikit-learn [BLB+13] library for the Random Forest and the OMP algorithm. We used a 60-20-20% split for train-validation-test data. Train data was used for training the base Random Forest and validation data for finding the hyper parameters using Bayesian optimization through the skopt library [HMS+18]. The train and validation data were then merged for complete retraining of the forest and selection. This seems counter-intuitive to not use the validation data separately for selecting the trees but our experiment have shown the best results were obtained without split for all techniques. For all techniques the results have been averaged over 10 runs and standard deviations are shown in plots.

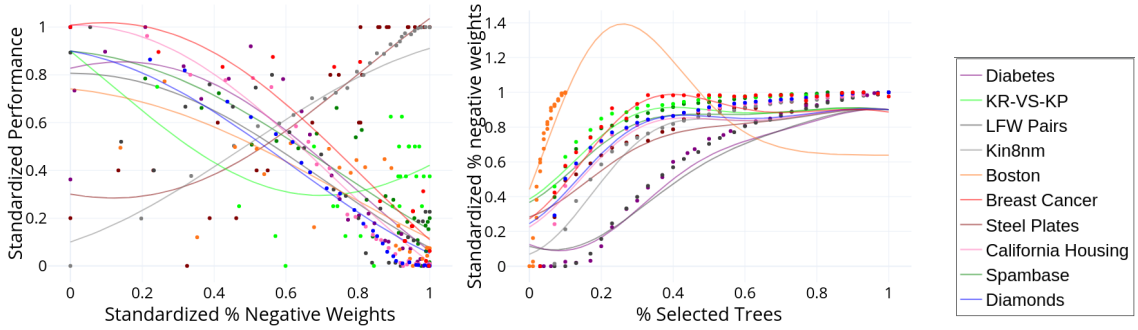
3.2 Results

We describe our results in this Section. We show that our method, and its ability to learn weights in particular, has excellent performance in some case but suffers harmful over-fitting in some scenarios. We remark an association between negativity of weights and bad performance which motivates the introduction of a non-negative constraint on the OMP weights. This heuristic allows our technique to not only achieve even better performance than the standard OMP version but also provides an early stopping criterion for advantageous selection of the pruned forest size.

3.2.1 Evaluating OMP based pruning

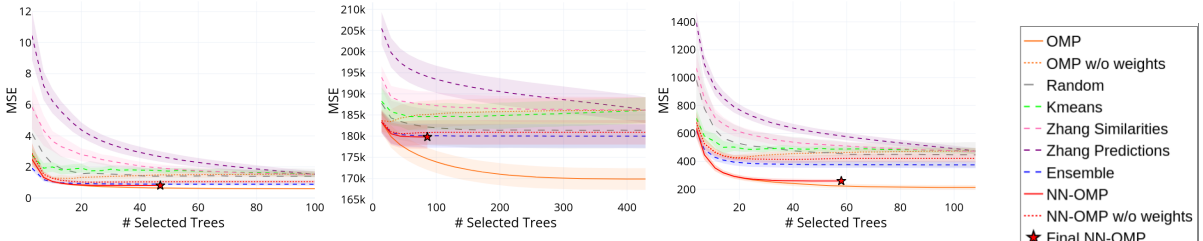
Results given in Figure 3 compare the performance of our OMP based pruning methods to other techniques. We first note that on some datasets, namely California Housing, Kin8nm, Steel Plates and KR-VS-VP, the OMP methods clearly outperform most other methods, reaching at its peaking value even better performance than the full Forest. On these datasets, we also clearly see the benefit of learning weights for the trees as the weighted version of OMP has better performance than its unweighted counterpart. The Ensemble method compete with OMP and shows excellent results on regression, as in the California Housing dataset. We remark that the Ensemble method is allowed to take multiple times the same tree in the forest, which is equivalent to setting a greater weight to that tree; this emphasize again the interest of using weights for the trees in the pruned forest. However, this Ensemble method performs extremely poorly on the binary classification tasks compared to the OMP based methods. The Zhang and Kmeans methods, although stable overall, struggle to perform even as good as random selection of trees in the forest and hence have worse performance than the OMP based pruning technique. A general description of the results is available in Table 2 where the best performance achieved by each method is shown, along with the corresponding number of trees.

In the other datasets, the unweighted version of our technique performs always *at least* as well as the



(a) Negative weigths effect on performance (b) Proportion of negative weights

Figure 1: Relation between forest size, negative weights and performance. In Figure 1a, only the pruned size of more than 10% than the base forest are considered. The inverse of the MSE is taken as performance measure for the regression dataset so that greater value means better performance. Performance values and percentages of negative weights are normalized using min-max scaling ($f(x) = (x - min)/(max - min)$) so that all values are comprised between 0 and 1 so they are comparable between datasets. Results are not represented for the Gamma dataset because it didn't have non negative weight values. The curves are used to facilitate the reading of the Figures and have been obtained by fitting for each dataset a SVM with rbf kernel and gamma value equal 1 on the raw obervations (the dots)



(a) Boston (b) Diamonds (c) Diabetes

Figure 2: Performance results on the **validation** data by the different methods with respect to the number of selected trees. One graph for each dataset that has shown bad results in Figure 3. Results are averaged over 10 runs and the light colour area around lines show the standard deviation.

	Ensemble		Kmeans		NN-OMP w/o weights		NN-OMP		OMP w/o weights		OMP		Random		Zhang Predictions		Zhang Similarities	
Diam.	3.032E+05	86	3.024E+05	143	**3.024E+05**	**86**	3.033E+05	86	3.025E+05	143	*3.087E+05*	*29*	3.025E+05	114	3.047E+05	143	3.032E+05	143
Diab.	3.431E+03	32	3.281E+03	36	3.317E+03	36	3.549E+03	36	3.324E+03	36	*3.607E+03*	*25*	3.303E+03	32	3.282E+03	36	**3.241E+03**	**32**
Kin.	1.892E-02	200	*2.024E-02*	*33*	1.921E-02	133	1.809E-02	133	1.931E-02	67	**1.776E-02**	**333**	2.002E-02	333	2.089E-02	333	2.017E-02	333
C. H.	2.187E-01	267	*2.449E-01*	*33*	2.239E-01	100	**2.180E-01**	**133**	*2.267E-01*	*33*	2.197E-01	133	2.390E-01	333	2.536E-01	333	2.452E-01	333
Bos.	1.267E+01	30	*1.278E+01*	*13*	**1.214E+01**	**33**	1.253E+01	33	1.247E+01	27	*1.293E+01*	*13*	1.253E+01	33	1.430E+01	33	1.283E+01	33
Sp. B.	94.27%	133	95.52%	167	*95.57%*	*100*	*95.59%*	*100*	95.56%	167	95.39%	133	**95.59%**	**167**	95.45%	333	95.46%	167
St. P.	98.69%	233	99.05%	267	*99.95%*	*67*	**99.95%**	**100**	*99.64%*	*67*	99.90%	333	*99.41%*	*67*	99.43%	167	98.92%	300
KR-KP	*98.22%*	*33*	99.00%	333	99.42%	100	99.39%	100	99.22%	100	**99.48%**	**100**	99.14%	267	99.14%	133	98.94%	333
B. C.	95.09%	100	***96.58%***	***33***	96.49%	67	**96.58%**	**67**	95.79%	133	95.35%	67	95.88%	300	*95.70%*	*33*	95.61%	333
LFW P.	*56.00%*	*67*	65.25%	333	**66.02%**	**333**	65.73%	233	65.32%	133	65.55%	167	65.98%	267	65.43%	333	65.27%	333
Gam.	*80.78%*	*3*	87.68%	33	87.75%	33	87.75%	33	87.75%	33	87.75%	33	**87.76%**	**33**	87.72%	33	87.68%	33

Table 2: Table representing the best performance atteined by each model along with the number of tree recquired for that performance. The maximum considered size of the pruned forest is 10% of the initial Random Forest. Best performance result on each dataset is **bold**, second best is underlined. Smallest number of tree is written in *italic*. Collocation of italic and bold or underlined means that the technique achieve excellent performance with the smallest number of tree. Score measures and acronyms for dataset are referenced in Table 1

other techniques but in some cases, namely the Diamonds, Boston, Diabetes and Breast Cancer datasets, the weights have an undisputable detrimental effect on the performance. We first remark that, in these cases, no technique seems able to give significantly better result than a simple random selection of trees. Second, these three datasets are very small in size and then we believe that this makes them extremely prone to over-fitting the weights as illustrated in Figure 2 which shows that the weighted forest performs extremely well on the Validation data, used for selecting trees.

3.2.2 Non-negativity of weights to prevent over-fitting

Even though the OMP based pruning technique has often excellent performance when selecting a small number of trees, it also produces bad results when the number of selected trees grows. This is extremely problematic as no particular hint is given during the selection phase of the trees which would allow us to design an early stopping criterion to set the maximum size for the pruned forest. Indeed, Figure 2 shows that the performance measure on the validation data doesn't seem to reach a particular plateau for that purpose.

Encouraged by the overall good performance of the unweighted version of OMP, we naturally investigate the weights values provided by OMP –instead of the trees themselves– in order to explain the poor behaviour of our technique. We notice, as depicted in Figure 1b that the *proportion* of trees with negative weight increases with the number of selected trees. This behavior has certainly detrimental effect on the performance of the pruned forest and we provide an explanation for binary classification where the labels are in $\{-1, +1\}$; the thought process is easily transferable to regression though.

It is reasonable to think that any of the pre-trained trees from the base forest should have *at least* better performance than random –even marginally– which means that mean accuracy for these trees should be over 0.5 in the case of binary classification. Yet, linearly combining trees with negative weights would be equivalent to reversing the vote of such trees, deliberately make it a *bad* classifier and use it for the final vote of the pruned forest. This counter-intuitive behaviour can mostly be explained by the fact that OMP chooses negative weights in order to perfectly match the training set labels, just as it has been designed to behave. This analysis is validated in Figure 1a where we clearly see that the performance tends to decrease when the proportion of negative weights increase. However, we must note that this trend is clearly not observed on Kin8nm and LFW Pairs. Note that only pruned forests with more than 10% of the base forest size are reported here. We choose to only report these points because the observations occurring before this threshold have naturally bad performance since they correspond to forest so small that they wouldn't have good performance anyway (See Figure 3).

This analysis lead us to the following conclusion: using a non-negative version of OMP[BEZ08] should prevent over-fitting. The Non-Negative version of OMP is similar to the one described in Algorithm 1 except that on line 3 where only positively correlated atoms can be selected and on line 5 where a non-negative least square regression is performed to obtain only non-negative coefficients. We remark that selecting only positively correlated atoms will eventually make NN-OMP unable to select new ones and hence prevent over-fitting by bounding the maximum size of the pruned forest. The results obtained with this algorithm are demonstrated in Figure 3 and Table 2 where we clearly see the benefit of our method in term of performance.

4 Conclusion

We presented in this paper an original method for reducing random forest's size. Our method uses a greedy algorithm, namely Orthogonal Matching Pursuit, on the predictions of each forest's tree to select a subset of trees that best approximate the real labels vector.

Our method not only selects a subset of trees, but also associate a coefficient to each selected tree. These coefficients are chosen so as to form a linear combination which best approximates the real labels vector and also they show really beneficial in some case, they may also lead to overfit in some other scenarios, as shown in an extensive set of experiments. We also provided theoretical insights and justifications for the proposed methodology.

The present work opens up several interesting research directions for future works. On the theoretical aspect, tying a tighter link between OMP and Gradient Boosting, and ensemble learning in general, becomes mandatory in order to better understand the underlying implications of our approach. A second possible research direction consists in providing custom-tailored theoretical guarantees for the proposed method based on frameworks such as Rademacher complexity. Finally, it would be interesting to quantify the interpretability gain of the forest extracted by non-negative OMP.

Acknowledgements

This work was partially supported by ANR project LIVES (ANR-15-CE23-0026-03), ANR project "Deep in France" (ANR16-CE23-0006), ERC project COVOPRIM (grant agreement ID: 788240).

References

[AP95] K.M Ali and M.J. Pazzani. On the link between error correlation and error reduction in decision tree ensemble. *Information and Computer Science, University of California, Irvine*, 1995.

[BAH12] Simon Bernard, Sébastien Adam, and Laurent Heutte. Dynamic random forests. *Pattern Recognition Letters*, 33(12):1580–1586, 2012.

[BCG+04] RK Bock, A Chilingarian, M Gaug, F Hakl, Th Hengstebeck, M Jiřina, J Klaschka, E Kotrč, P Savickỳ, S Towers, et al. Methods for multidimensional event classification: a case study using images from a cherenkov gamma-ray telescope. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 516(2-3):511–528, 2004.

[BEZ08] Alfred M Bruckstein, Michael Elad, and Michael Zibulevsky. Sparse non-negative solution of a linear system of equations is unique. In *2008 3rd International Symposium on Communications, Control and Signal Processing*, pages 762–767. IEEE, 2008.

[BK10] Gavin Brown and Ludmila I. Kuncheva. "good" and "bad" diversity in majority vote ensembles. In Neamat El Gayar, Josef Kittler, and Fabio Roli, editors, *Multiple Classifier Systems*, pages 124–133, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[BKW80] David A Belsley, Edwin Kuh, and Roy E Welsch. *Regression diagnostics: Identifying influential data and sources of collinearity*, volume 571. John Wiley & Sons, 1980.

[BLB+13] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[Bre01] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[Bus98] Massimo Buscema. Metanet*: The theory of independent judges. *Substance use & misuse*, 33(2):439–461, 1998.

[CL98] Lorrie Faith Cranor and Brian A LaMacchia. Spam! *Communications of the ACM*, 41(8):74–83, 1998.

[CMS19] Corinna Cortes, Mehryar Mohri, and Dmitry Storcheus. Regularized gradient boosting. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 5449–5458. Curran Associates, Inc., 2019.

[CNMCK04] Rich Caruana, Alexandru Niculescu-Mizil, Geoff Crew, and Alex Ksikes. Ensemble selection from libraries of models. In *Proceedings of the twenty-first international conference on Machine learning*, page 18. ACM, 2004.

[Cor96] Peter I Corke. A robotics toolbox for matlab. *IEEE Robotics & Automation Magazine*, 3(1):24–32, 1996.

[Dia] Diamonds dataset. `https://www.kaggle.com/shivam2503/diamonds`.

[DMA97] Geoff Davis, Stephane Mallat, and Marco Avellaneda. Adaptive greedy approximations. *Constructive approximation*, 13(1):57–98, 1997.

[EHJ+04] Bradley Efron, Trevor Hastie, Iain Johnstone, Robert Tibshirani, et al. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004.

[FGE15] Khaled Fawagreh, Mohamad Medhat Gaber, and Eyad Elyan. On extreme pruning of random forest ensembles for real-time predictive applications. *arXiv preprint arXiv:1503.04996*, 2015.

[FME15] Khaled Fawagreh, Mohamad Medhat Gaber, and Eyad Elyan. On Extreme Pruning of Random Forest Ensembles for Real-time Predictive Applications. *arXiv e-prints*, page arXiv:1503.04996, Mar 2015.

[Fri01] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[GV05] Rémi Gribonval and Pierre Vandergheynst. On the exponential convergence of matching pursuits in quasi-incoherent dictionaries. *IEEE Transactions on Information Theory*, 52(1):255–261, 2005.

[HMBLM08] Gary B Huang, Marwan Mattar, Tamara Berg, and Eric Learned-Miller. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. 2008.

[HMS+18] Tim Head, Louppe MechCoder, Iaroslav Shcherbatyi, et al. scikit-optimize/scikit-optimize: v0. 5.2, 2018.

[HYXL07] Qinghua Hu, Daren Yu, Zongxia Xie, and Xiaodong Li. Eros: Ensemble rough subspaces. *Pattern recognition*, 40(12):3728–3739, 2007.

[KS12] Vrushali Y Kulkarni and Pradeep K Sinha. Pruning of random forest classifiers: A survey and future directions. In *2012 International Conference on Data Science & Engineering (ICDSE)*, pages 64–68. IEEE, 2012.

[LRK+18] Francesco Locatello, Anant Raj, Sai Praneeth Karimireddy, Gunnar Rätsch, Bernhard Schölkopf, Sebastian U. Stich, and Martin Jaggi. On matching pursuit and coordinate descent, 2018.

[MBBF00] Llew Mason, Jonathan Baxter, Peter L Bartlett, and Marcus R Frean. Boosting algorithms as gradient descent. In *Advances in neural information processing systems*, pages 512–518, 2000.

[MSW95] Olvi L Mangasarian, W Nick Street, and William H Wolberg. Breast cancer diagnosis and prognosis via linear programming. *Operations Research*, 43(4):570–577, 1995.

[PB97] R Kelley Pace and Ronald Barry. Sparse spatial autoregressions. *Statistics & Probability Letters*, 33(3):291–297, 1997.

[PRK93] Yagyensh Chandra Pati, Ramin Rezaiifar, and Perinkulam Sambamurthy Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Proceedings of 27th Asilomar conference on signals, systems and computers*, pages 40–44. IEEE, 1993.

[Rok09] Lior Rokach. Collective-agreement-based pruning of ensembles. *Computational Statistics & Data Analysis*, 53(4):1015–1026, 2009.

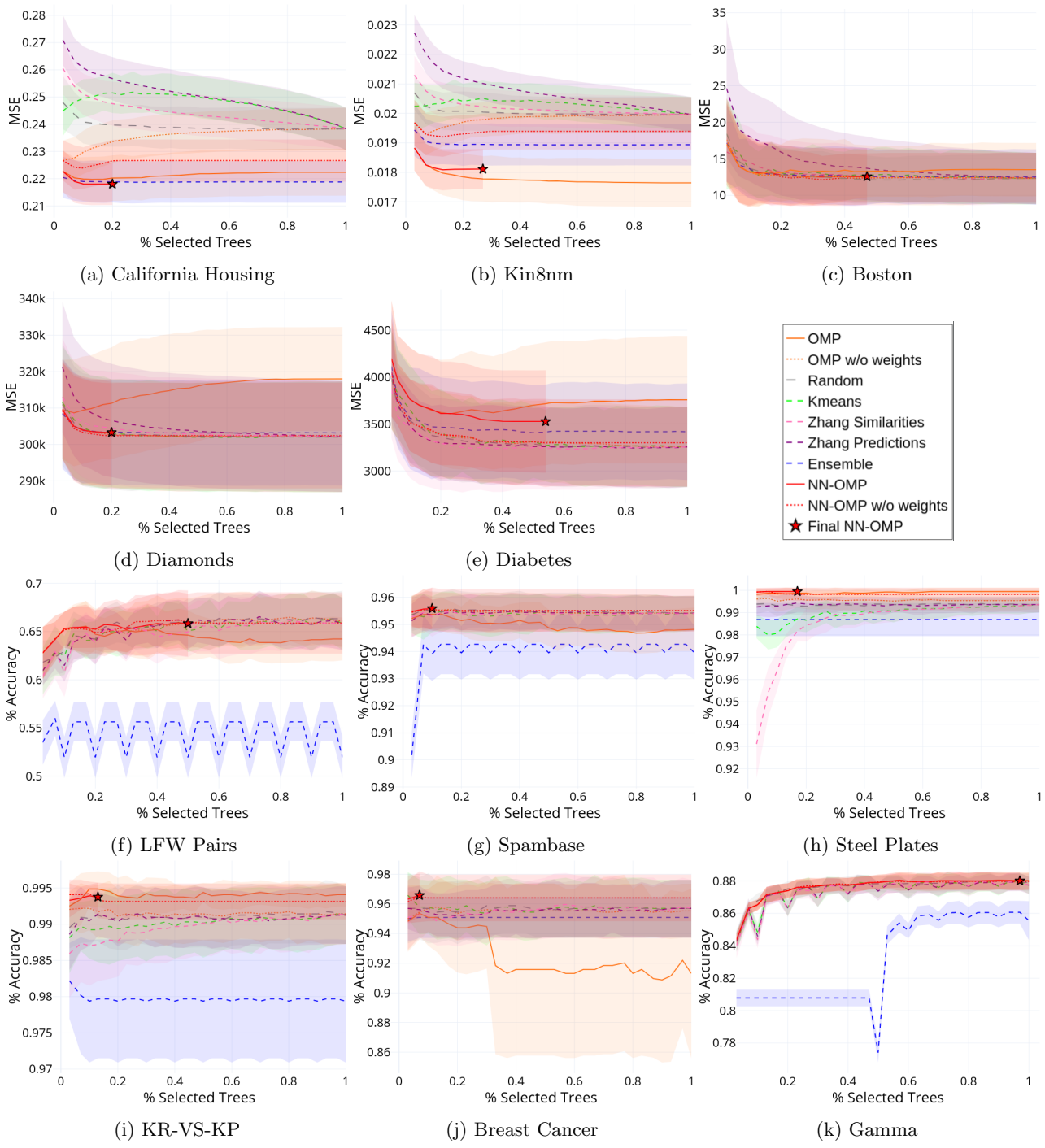
[Sha87] Alen D Shapiro. *Structured induction in expert systems*. Addison-Wesley Longman Publishing Co., Inc., 1987.

[Tro04] Joel A Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Transactions on Information theory*, 50(10):2231–2242, 2004.

[WZ09] Minghui Wang and Heping Zhang. Search for the smallest random forest. *Statistics and its Interface*, 2(3):381–388, 2009.

[YLLL12] Fan Yang, Wei-hang Lu, Lin-kai Luo, and Tao Li. Margin optimization based pruning for random forest. *Neurocomputing*, 94:54–63, 2012.

[ZW09] Heping Zhang and Minghui Wang. Search for the smallest random forest. *Statistics and its Interface*, 2(3):381, 2009.



(a) California Housing (b) Kin8nm (c) Boston

(d) Diamonds (e) Diabetes

(f) LFW Pairs (g) Spambase (h) Steel Plates

(i) KR-VS-KP (j) Breast Cancer (k) Gamma

Figure 3: Performance results on the **test** data by the different methods with respect to the number of selected trees. One graph for each tested dataset. Results are averaged over 10 runs and the light colour area around lines show the standard deviation. See Table 1 for the size of the base forest