Global Refinement of Random Forest

Shaoqing Ren Xudong Cao Yichen Wei Jian Sun
University of Science and Technology of China Microsoft Research
sqren@mail.ustc.edu.cn {xudongca,yichenw,jiansun}@microsoft.com

Abstract

Random forest is well known as one of the best learning methods. In spite of its great success, it also has certain drawbacks: the heuristic learning rule does not effectively minimize the global training loss; the model size is usually too large for many real applications. To address the issues, we propose two techniques, global refinement and global pruning, to improve a pre-trained random forest. The proposed global refinement jointly relearns the leaf nodes of all trees under a global objective function so that the complementary information between multiple trees is well exploited. In this way, the fitting power of the forest is significantly enhanced. The global pruning is developed to reduce the model size as well as the over-fitting risk. The refined model has better performance and smaller storage cost, as verified in extensive experiments.

1. Introduction

Random forest [4] is one of the most popular learning methods and has many ideal properties: 1) it is simple to understand and implement; 2) it is strong in handling non-linearity and outliers; 3) it is friendly to parallel training and large data; and 4) it is fast in testing. Recently, it has proven extremely successful on important applications in data mining [33] and computer vision [31, 14, 30].

In spite of its great success, random forest has certain insufficiency from both theoretical and practical viewpoints. Theoretically, the heuristic learning of random forest is sub-optimal in terms of minimizing training error. Specifically, each individual tree is learnt independently and greedily. Such learning does not fully utilize complementary information among different trees.

Practically, for complex real problems [31, 14, 12, 10], deep trees are usually required to fit the training data well. This results in high storage cost, which is a serious issue especially for embedded devices such as mobile phone or Kinect. While tree pruning can reduce the tree size, existing methods [25, 19] are independently performed on individual trees and could degrade the performance of random forest.

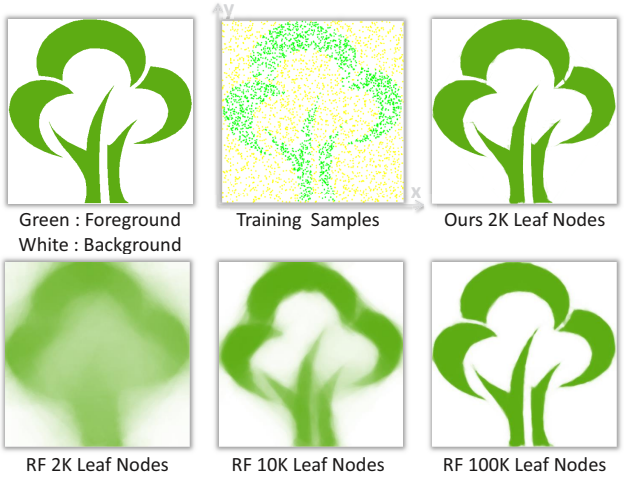

Figure 1. A toy classification task. From left to right, the first row shows the groundtruth label map, training data points, and the probability map predicted by our refined forest. The second row shows the probability map predicted by standard random forest (RF) trained with various depth ($D_{\max} = 5, 8, 12$). Our refined forest can clearly separate the two classes using a smaller number of leaf nodes. In all cases, we use 100 trees in the forest.

To address the above problems, we propose a simple and effective method to refine a pre-trained random forest. We notice that the learning and prediction of random forest is inconsistent: the learning of individual trees is independent but the prediction averages all trees' outputs. The loss functions implied from these two processes are actually different. This limits the fitting power of random forest. To alleviate such inconsistency, we discard the old values stored in all tree leaves of a pre-trained random forest and relearn them through a "global" refinement: *all tree leaves are simultaneously optimized by explicitly minimizing a global loss function defined on all training samples, according to the*

averaging prediction rule of random forest.

The proposed global refinement can be efficiently solved with a linear support vector classification/regression machine. As a result, the complementary information between trees is exploited and the fitting power is significantly improved.

The global optimization in training might cause over-fitting for a large number of deep trees (therefore a huge number of tree leaves). To reduce the risk of over-fitting as well as the model size, we propose a global pruning method which alternates between refining all tree leaves and merging the insignificant leaves, similar to the sparse approximation [3]. In this way, the model size is significantly reduced and the generalization capability is usually improved. A toy example is shown in Figure 1.

In our various experiments, the improved random forest achieves better accuracy and smaller model size, compared to standard random forest and some state-of-the-art variants. The strong results verify the effectiveness and practicability of our approach.

Our formulation is also applicable to other ensemble tree-based models. The preliminary results on boosting trees [13] and alternating decision forests [29, 28] are encouraging.

2. Related works

Random forest. Random forest [4] is an ensemble of trees that are trained independently. Certain randomness is injected to decorrelate the trees. To make a prediction, random forest combines the predictions of all individual trees by *averaging*, which is the key for generalization [8].

The randomness can be injected by randomly sampling a subset of parameters for training a splitting node; or randomly sampling a subset of training data for learning an individual tree (i.e. *bagging*). Deep trees are the main source of the strength of random forest. Although reducing the randomness could also improve the strength of a single tree, it is preferable to increase the depth of trees because a certain randomness is needed to ensure the complementarity among different trees.

Random forest as "visual codebook". Extremely Randomized Clustering Forests (ERC-Forests) [22] treats random forest as visual codes describing images, followed by SVM as classifier in visual recognition task. Thanks to the efficiency of random forest, ERC-Forests out-performs k-means based coding in both accuracy and computational efficiency.

Opposite to treating random forest as codec. Our motivation is to improve general random forest. Therefore our designs bear a few novel contributions from the viewpoint and fully consider the formulation and structure of random forest.

Variants of random forest. As pointed out in [4], the generalization error of random forest is determined by the average strength and correlation of individual trees: the stronger the strength and the lower the correlation, the better the performance. To enhance the strength of individual trees, [20] and [24] propose new criteria to learn stronger splitting nodes; [21, 17, 18] propose some new features to encode rich and discriminative context information in computer vision tasks. Injecting more randomness also helps, e.g., randomly choosing purity function from multiple candidates [27] or randomly deciding how many times of feature evaluation [2]. Combining random forest with graphical model makes another great work [23].

Recently, a new line of research focuses on constructing complementary trees. Notable works include dynamic random forest [1], alternating decision forests (ADF) [29] and alternating regression forests (ARF) [28]. All these works exploit the reweighting/resampling scheme of gradient boosting to induce complementary trees. Unlike these works, we focus on utilizing the complementary information of a pre-trained random forest.

3. Our approach

We start by introducing notations for a tree and reformulating the tree's prediction rule into a compact form to facilitate our new formulation later.

- The indicator vector $\phi(x)$: the structure of a tree can be considered as a mapping function that maps an input data point x to an indicator vector $\phi(x)$, which is binary and has the same dimension of number of leaf. Each dimension indicates whether the corresponding leaf node contains the input data point or not (1 or 0).
- The leaf matrix w: each leaf node stores a vector for prediction (*e.g.*, a posterior distribution for classification or continuous values for regression). We pack all leaf vectors into a matrix w, with each column corresponding to a leaf node.

The prediction of a tree can thus be expressed as

$$y = w\ \phi(x), \tag{1}$$

where the vector y is the predicted output. This compact form is the building block of our approach.

3.1. Global refinement of random forest

We first reformulate the learning of the leaf vectors of random forest from the viewpoint of loss function minimization, from which we can clearly motivate our global refinement formulation. We start from a single tree.

Reformulation of a single tree. We cast the rule for learning the leaf vectors of a single tree into the following objective function,

$$\min_{w} \ \frac{1}{N}\sum_{i=1}^{N} l(y_i, \hat{y}_i) \qquad (2)$$
$$s.t. \ \ y_i = w\ \phi(x_i), \ \ \forall i \in [1, N],$$

where N is the number of training samples and $l(y_i, \hat{y}_i)$ is the loss function defined on the prediction y_i and the groundtruth $\hat{y}_i$. For classification, we use the hinge loss to achieve maximum-margin properties[1]. For regression, we use the mean square error to measure loss.

As the training data are partitioned into different leaf nodes, the optimization problem in (2) is degenerated into a series of independent local optimization problems. By "*local*", we mean that the leaf vector is learnt only from the training data falling into the leaf node. The local optimization problem is easy to solve. For classification, the optimal leaf vector is the class distribution estimated from samples in the leaf. For regression, the optimal leaf vector is the mean value of the samples in the leaf.

Reformulation of random forest. Similarly, we present the reformulated rule of random forest in the task of learning the leaf vectors.

$$\min_{w} \ \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T} l(y_i^t, \hat{y}_i) \qquad (3)$$
$$s.t. \ \ y_i^t = w_t\ \phi_t(x_i), \ \ \forall i \in [1, N], \ \ \forall t \in [1, T],$$

where T is the number of trees, $\phi_t(x)$ and w_t are the indicator vector and the leaf matrix of the t^{th} tree, y_i^t is the prediction of the t^{th} individual tree.

In (3), the leaf matrices $[w_1, w_2, ..., w_T]$ from all the trees are independent from each other. The optimization problem is degenerated into T independent subproblems, each corresponding to an instance of (2). Therefore, learning the random forest's leaves is reduced to learning each leaf node independently.

Our formulation of refining random forest. From the reformulation in (3), we can identify the key problem: the loss function implied by random forest learning is different from the ideal loss function corresponding to the random forest prediction. The two loss functions are compared as follows.

$$\text{The loss of RF learning:} \quad \frac{1}{T}\sum_{t=1}^{T} l(y_i^t, \hat{y}_i) \qquad (4a)$$

$$\text{The ideal loss:} \quad l(y_i, \hat{y}_i), \ y_i = \frac{1}{T}\sum_{t=1}^{T} y_i^t \qquad (4b)$$

We note that the latter is the ideal loss function because $y_i = 1/T\sum_{t=1}^{T} y_i^t$ is the final prediction of random forest, by averaging trees' predictions.

The inconsistency between the learning (training) and prediction (testing) is clearly suboptimal. This in turn results in limited fitting power. Based on this insight, we propose to refine the leaf nodes of random forest by adopting the ideal loss function in Eq. (4b). Our formulation of global refinement is obtained by plugging the ideal loss function into the reformulation of random forest in Eq. (3).

For notation clarify, we pack the indicator vector $\phi_t(x)$ and the leaf matrix w_t of individual trees into a large indicator vector $\Phi(x)$ and the leaf matrix W as

$$\Phi(x) = [\phi_1(x); ...; \phi_t(x); ...; \phi_T(x)], \qquad (5)$$
$$W = [w_1, ..., w_t, ..., w_T].$$

Similar to the prediction of a single tree in Eq. (1), the prediction of the random forest can be expressed as

$$y = W\ \Phi(x), \qquad (6)$$

where the vector y is the prediction of random forest.

With Eq. (5) and (6), we present the global refinement of random forest in the following form.

$$\min_{W} \ \frac{1}{2}\|W\|_F^2 + \frac{C}{N}\sum_{i=1}^{N} l(y_i, \hat{y}_i) \qquad (7)$$
$$s.t. \ \ y_i = W\ \Phi(x_i), \ \ \forall i \in [1, N].$$

We add an additional L2 regularization term $\frac{1}{2}\|W\|_F^2$ on leaf vectors to reduce the risk of over-fitting. The parameter C controls the tradeoff between the L2 regularization and the loss of training data.

The objective function in Eq. (7) has the same form as in the support vector machine [32], thus it can be solved by convex optimization with global optimum. The optimization can be greatly accelerated by exploiting the sparsity of the indicator vectors. In our implementation, we use liblinear library [11] which can handle this sparse optimization very well.

[1] For the multi-class classification, we use the well-known one-vs-the-rest approach [32].

In the global refinement, we do not change the structure of trees, but only refine the vectors stored in tree leaves. The usage of the refined random forest is the same as before in testing.

Discussions. In standard random forest, the trees are independent during learning. No information is shared between trees. The main benefit of our approach is that it effectively utilizes complementary information among multiple trees. Leaf nodes from different trees are now related as they collaboratively predict the same sample datum. The fitting power is substantially improved by global optimization.

As the complementary effect among different trees is the key for refinement, this suggests that performance may be further improved by inducing more complementary information into the random forest, such as increasing the number of trees, or decreasing the correlation between different trees. Experiments in Section 4.4 shows that these strategies are indeed effective.

3.2. Global pruning

In real complex problems, usually a large number of deep trees are necessary. With a huge number of tree leaves(typically millions), above global refinement in training might cause over-fitting. Besides the regularization defined in Eq. (7), another regularization is necessary to alleviate over-fitting, that is, limiting the number of total tree leaves.

Existing tree pruning methods [25, 19] independently merge the leaf nodes of each individual trees. Similar to local learning, this local pruning is suboptimal as it does not consider the global loss function in Eq. (7).

We propose a more effective global pruning method. It iteratively merges insignificant tree leaves using global optimization, as follows.

1. Optimizing the leaf vectors according to Eq. (7). The parameter C is determined by cross validation.

2. Pruning insignificant leaves: two adjacent leaves are merged into one new leaf if the norm of their leaf vectors are close to zeros. Specifically, for each pair of adjacent leaves, we first compute the summation of the l2-norm of their leaf vectors to measure the significance. Then we merge a certain percentage of least significant pairs into new leaves.

3. Updating the indicator vectors according to the new tree structures. The update is very efficient because we only need to remove the indicator value of the pruned old leaves and add the indicator value of emerged new leaves. The indicator value of a new leaf is the summation of the indicator values of the merged leaves.

4. Repeating step 1, 2 and 3 until satisfying certain stopping conditions, *e.g*. the model size is smaller than a predefined size, or the accuracy achieves the best on a validation set.

In this way, all leaf nodes are jointly considered to determine the least important ones for removal. This is better than the pruning of individual trees independently. The tree structures are kept updated, ensuring that the current leaf vectors are always optimal and current pruning is well informed. In experiments, we found that the proposed global pruning can significantly reduce model size meanwhile still retain fairly good performance. The testing accuracy usually improves when the pruning is relatively moderate which indicates the global pruning reduces over-fitting.

4. Experiments

We firstly evaluate the proposed approach on standard machine learning tasks (Section 4.1) and two real applications, one for classification (Section 4.2) and one for regression (Section 4.3). We then study the impacts of important parameters in standard random forest and our refinement in Section 4.4.

Experiment settings In Section 4.1, 4.2 and 4.3, our refined random forest is compared with the original forest before refinement (referred to as RF), and two state-of-the-art variants: Alternating Decision Forests (ADF) [29] and Alternating Regression Forests (ARF) [28]. The two variants integrate a gradient boosting like training process into the random forest learning and show superior performance. For these methods we use the same number of trees and depth as in the standard/refined forest. Their other parameters are set as recommended in the authors' papers.

During iterative pruning, we found that the less leaves pruned in each iteration, the better the accuracy but the slower the training. As a tradeoff, we empirically prune 10% of leaves in each iteration until the stopping conditions are met. The optimal parameter C in global pruning and refinement are decided by cross validation. The candidate values of parameter C are $[10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1]$.

To investigate the accuracy/memory trade off, we specify two different stopping conditions for pruning: 1) when its accuracy achieves its best and further pruning would hurt performance; 2) when its accuracy is comparable to the o-

Dataset	#Train	#Test	#Feature	#Classes or #TargetDim
(c) letter	15000	5000	16	26
(c) usps	7291	2007	256	10
(c) Char74k	66707	7400	64	62
(c) MNIST	60000	10000	784	10
(c) covtype	348607	232405	54	7
(r) abalone	2506	1671	8	1
(r) ailerons	7154	6596	40	1
(r) cpusmall	4915	3277	12	1
(r) cadata	12384	8256	8	1
(r) deltaelevators	5710	3807	6	1

Table 1. The properties of the standard machine learning datasets used in Section 4.1. The top five are classification (c) and bottom five are regression (r).

riginal random forest. We call the first accurate version (refined-A) which corresponds to the optimal pruning point, and we call the second economic version (refined-E) which corresponds to over-pruning but obtains a much small model.

We use standard performance evaluation metrics: error rate for classification and Root Mean Square Error(RMSE) for regression unless otherwise specified.

4.1. Standard machine learning tasks

We test 10 standard machine learning tasks: 5 for classification, and 5 for regression. The datasets are provided by [5][2] and [29]. The properties of the datasets are summarized in Table 1.

We follow the experiment settings in [28, 29]. The number of trees is set to 100. The number of random features tested in each node is set to the square root of the feature dimension, as recommended in [4]. We set the maximum tree depth $D_{\max}$ as either 10, 15, or 25, depending on the size of the training data. The minimum sample number for split is set to 5. For datasets without standard training/testing split, we randomly split the dataset into 60% for training and 40% for testing. To decrease the statistical fluctuation due to randomness in the forest learning, we report the mean error and standard deviation of 10 rounds experiments.

The results are presented in Table 2. In terms of accuracy, our accurate version significantly improves the standard random forest on all classification tasks and 3 out of 5 regression tasks. It also consistently outperforms the state-of-the-art ADF/ARF. In terms of model size, our accurate version is 2∼30 times smaller than RF, and our economic version is 15∼160 times smaller without any performance loss relative to RF. We note that the model size of ADF/ARF is similar to the standard random forest.

[2]http://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/

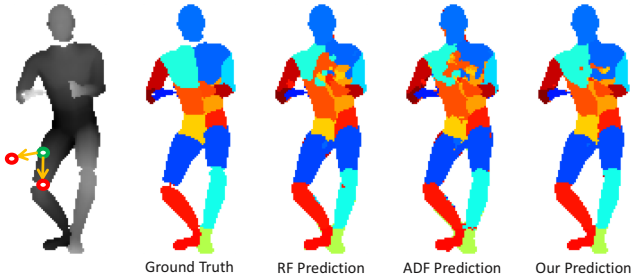

Figure 2. Visual results of Kinect body part classification. The left figure illustrates the pixel-difference feature on the depth image. The remaining figures show the ground-truth and the results of different methods.

Such strong results on diverse tasks clearly demonstrate the effectiveness of our approach. Below we study large scale and real world applications.

4.2. Application I: Kinect body part classification

One of the most successful application of random forest is human body part recognition [31]. Since no public dataset is provided in [31], we use a recent public dataset in [9]. It contains 2,500 depth images at 320*240 resolution, with 2,000 for training and 500 for testing. Each depth image is pixel-wise labelled into 20 classes, including 19 body parts and the background.

In our training, from each depth image we randomly generate 400 samples (20 samples per class). There are 800K training samples and 200K testing samples in total. To train a strong model, we use 100 trees and set the max tree depth to 25. We use standard scale-invariant pixel-difference features [31] (see Figure 2) and set the number of random features on each split node as 500.

Results in Table 3 show that our accurate version can significantly reduce error and moderately reduce the model size. Yet, the large model size can still be a serious concern. Our economic version can compress the model size by 40 times and still retain a comparable accuracy relative to the standard random forest. This is a huge benefit for real applications.

Method	RF	ADF	refined-A	refined-E
Error (%)	8.10	7.57	**4.55**	7.96
Model size(MB)	290.4	369.5	177.6	**6.80**

Table 3. Results on Kinect body part classification task.

We also report the training time on a single-core ma-

Dataset	Performance (Error)					Compression Ratio	
	Error Scale	RF	ADF/ARF	refined-A	refined-E	refined-A	refined-E
(c) letter	10^{-2}	4.50±0.13	3.76±0.14	**2.98±0.15**	4.33±0.08	2.33	**30.32**
(c) usps	10^{-2}	6.21±0.21	5.60±0.16	**5.10±0.10**	5.69±0.15	2.86	**15.14**
(c) Char74k	10^{-2}	18.3±0.15	16.9±0.16	**15.4±0.10**	18.0±0.09	1.70	**37.04**
(c) MNIST	10^{-2}	3.14±0.04	2.73±0.05	**2.05±0.02**	2.95±0.03	6.29	**76.92**
(c) covtype	10^{-2}	16.4±0.10	15.3±0.11	**4.11±0.04**	15.6±0.08	1.68	**166.67**
(r) abalone		2.11±0.05	**2.10±0.03**	**2.10±0.01**	2.11±0.03	12.65	**16.67**
(r) ailerons	10^{-4}	2.01±0.01	1.98±0.01	**1.75±0.02**	1.95±0.02	33.13	**124.82**
(r) cpusmall		3.15±0.05	2.95±0.04	**2.90±0.05**	3.02±0.03	22.73	**66.53**
(r) cadata	10^{4}	5.50±0.05	5.40±0.05	**5.05±0.06**	5.36±0.05	36.14	**62.50**
(r) deltaelevators	10^{-3}	**1.46±0.04**	**1.46±0.02**	**1.46±0.03**	1.46±0.03	**37.04**	**37.04**

Table 2. Results on standard machine learning datasets. Left: errors of compared methods. Right: model size reduction by our method relative to standard random forest (RF), measured in compression ratio.

Method	Error (MAE)	Model size (MB)
[16]	**4.20**	-
refined-A	**4.43**	1.61
refined-E	4.83	0.18
RF	6.01	9.77
ARF	5.73	9.66
[6]	6.07	-
[15]	4.18	-

Table 4. Results of age estimation.

chine. It takes around 200K seconds to train the standard random forest. During iterative pruning, the dominated time cost is from the first step, i.e., determining the optimal C and computing the refined leaf vectors. It takes 8K seconds per iteration. For our accurate and economic versions, the numbers of iterations are 5 and 40 respectively.

4.3. Application II: human age regression

We use the dataset MORPH Album 2 (academic) [26]. It contains 55,134 images of 13,618 subjects. Ages range from 16 to 77 with a median of 33. Following standard literature, we use Mean Absolute Error (MAE) as the performance measurement.

We follow the experiment settings of the recent state-of-the-art work [16]. We randomly sample 10,000 images for training and use the remaining for testing. For face representation, we use high dimensional Local Binary Patterns (LBP) features [7] and compress the dimensionality to 2,000 by PCA. We choose a strong parameter setting for standard forest: 1,000 trees, max depth 15 and 500 random splits. Under this setting, the performance of standard forest almost saturates.

Table 4 compares several random forest based methods as well as state-of-the-art ones on this task. Similarly, our refinement is significantly better than both standard random forest and ARF in terms of both accuracy and model size. We notice [16] achieves the best result by using task-specific hierarchical age estimation and biological inspired model on 68 landmarks. Given these considerations, our method still achieves very competitive performance.

For reference, we also report the results of other two state-of-the-arts in [6] and [15]. Both methods conduct experiments on selected subsets[3].

4.4. Analysis of random forest parameters

As analyzed in Section 3, our global refinement benefits from the complementary information between different trees. In this section, we investigate several important parameters that affect the complementary property. We use the well known MNIST dataset and the same parameter settings unless otherwise noted in this section as in Section 4.1. To isolate global refinement for study, we do not apply global pruning in this section.

Number of random features tested on a split node. Figure 3 (left) shows the error curves of standard random forest and our refinement by varying this parameter. The optimal value for standard random forest is around 100, but for our refinement it is 10. This means that the refinement benefits from a more random feature selection. Testing a small number of features also speeds up the training.

Bagging ratio is the proportion of the whole training data used in training each tree. Figure 3 (right) shows the error curves by varying this parameter. It shows that standard

[3]As the distributions of ethnic groups are unbalanced in MORPH-II dataset, [6] selected a subset which contains 5,123 face images of Caucasian descent only and used 80% for training. [15] selected a subset of 21,000 images with balanced ethnics and used 10,000 images for training.

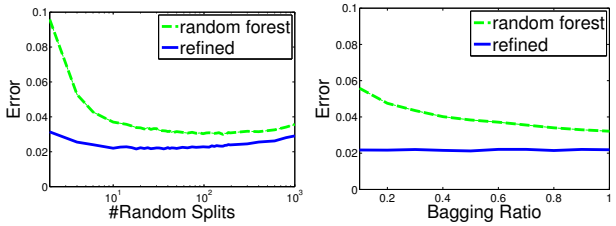

Figure 3. The impacts of randomness: the number of random splits (left) and bagging ratio (right).

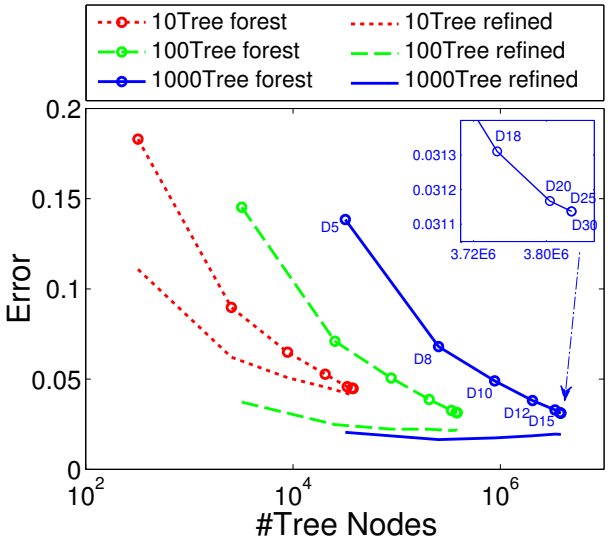

Figure 4. Tree number vs. depth. Provided a fixed number of trees, we plot the corresponding curve by varying the maximum depth ($D_{\max} = 5, 8, 10, 12, 15, 18, 20, 25, 30$), resulting in different number of leaf nodes. The data points for depth = $(18, 20, 25, 30)$ are almost overlapped (as shown in the zoom-in sub figure) because most branches do not reach very large depth due to other stopping criterions such as the minimum number of samples and the maximum purity.

random forest requires more data per tree for better performance, but our refinement is less sensitive to this parameter. This means that training with refinement could be faster with less data. In all experiments in Section 4.1, 4.2, and 4.3, we set this ratio to 1 so that all comparisons are actually in favor of random forest.

Tree number vs. depth. When the total number of leaf nodes (model size) is fixed, there is a trade-off between tree number and depth. The results using different number of trees are shown in Figure 4. Clearly, more trees and more leaf nodes lead to better performance but a less obvious observation is that the standard random forest prefers a small number of deep trees while our refinement prefers more shallow trees, and the relative performance improvement increases as the number of trees increases.

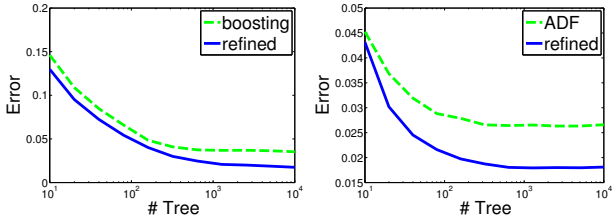

Figure 5. Our refinement also improves boosting trees and ADF on MNIST task, using different number of trees.

As both high randomness and large tree number bring in rich complementary information, the above studies imply that more complementary information leads to better performance. This evidences that we effectively exploit the complementary information among multiple trees in the global refinement.

5. Conclusions and future works

In this work, we presented a global refinement algorithm to improve the fitting power of a pre-trained random forest. We also developed a global pruning algorithm to reduce the over-fitting risk as well as the model size. Such benefits are almost free because the refining/pruning optimization is efficient and the testing speed is as fast as before.

Besides random forest, our formulation can also be applied to other ensemble tree based models. This is investigated by a preliminary experiment on MNIST dataset. In Figure 5, we show that similar performance improvement is also observed by refining the boosting trees and alternating decision forests (ADF) [29]. For boosting trees, we use a shallow depth 5 as in the common practice. Other parameters are the same as in Section 4.1. Interestingly, the trees in such models are no longer independent during training and it is less obvious how complementary the trees are. Our initial results show that a global optimization over the entire training data is still helpful. This is certainly an interesting direction for future further study.

References

[1] S. Bernard, S. Adam, and L. Heutte. Dynamic random forests. *Pattern Recognition Letters*, 33(12):1580–1586, 2012. 2

[2] S. Bernard, L. Heutte, and S. Adam. Forest-rk: A new random forest induction method. In *Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence*, pages 430–437. Springer, 2008. 2

[3] T. Blumensath and M. E. Davies. Iterative thresholding for sparse approximations. *Journal of Fourier Analysis and Applications*, 14(5-6):629–654, 2008. 2

[4] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001. 1, 2, 5

[5] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm. 5

[6] K.-Y. Chang, C.-S. Chen, and Y.-P. Hung. Ordinal hyperplanes ranker with cost sensitivities for age estimation. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 585–592. IEEE, 2011. 6

[7] D. Chen, X. Cao, F. Wen, and J. Sun. Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3025–3032. IEEE, 2013. 6

[8] A. Criminisi, J. Shotton, and E. Konukoglu. Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. *Foundations and Trends® in Computer Graphics and Vision*, 7(2-3):81–227, 2011. 2

[9] M. Denil, D. Matheson, and N. de Freitas. Consistency of online random forests. In *International Conference on Machine Learning (ICML)*, pages 1256C–1264, 2013. JMLR W&CP 28 (3): 1256C1264, 2013. 5

[10] P. Dollár and C. L. Zitnick. Structured forests for fast edge detection. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1841–1848. IEEE, 2013. 1

[11] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874, 2008. 3

[12] G. Fanelli, J. Gall, and L. Van Gool. Real time head pose estimation with random regression forests. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 617–624. IEEE, 2011. 1

[13] J. H. Friedman. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, pages 1189–1232, 2001. 2

[14] J. Gall and V. Lempitsky. Class-specific hough forests for object detection. In *Decision Forests for Computer Vision and Medical Image Analysis*, pages 143–157. Springer, 2013. 1

[15] G. Guo and G. Mu. Simultaneous dimensionality reduction and human age estimation via kernel partial least squares regression. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 657–664. IEEE, 2011. 6

[16] C. O. Hu Han and A. K. Jain. Age estimation from face images: Human vs. machine performance. In *The 6th IAPR International Conference on Biometrics (ICB)*, 2013. 6

[17] P. Kontschieder, S. R. Bulò, A. Criminisi, P. Kohli, M. Pelillo, and H. Bischof. Context-sensitive decision forests for object detection. In *Advances in neural information processing systems*, pages 431–439, 2012. 2

[18] P. Kontschieder, P. Kohli, J. Shotton, and A. Criminisi. Geof: Geodesic forests for learning coupled predictors. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 65–72. IEEE, 2013. 2

[19] M. Mehta, J. Rissanen, R. Agrawal, et al. Mdl-based decision tree pruning. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, volume 95, pages 216–221, 1995. 1, 4

[20] B. H. Menze, B. M. Kelm, D. N. Splitthoff, U. Koethe, and F. A. Hamprecht. On oblique random forests. In *Machine Learning and Knowledge Discovery in Databases*, pages 453–469. Springer, 2011. 2

[21] A. Montillo, J. Shotton, J. Winn, J. E. Iglesias, D. Metaxas, and A. Criminisi. Entangled decision forests and their application for semantic segmentation of ct images. In *Information Processing in Medical Imaging*, pages 184–196. Springer, 2011. 2

[22] F. Moosmann, B. Triggs, and F. Jurie. Fast discriminative visual codebooks using randomized clustering forests. In *Twentieth Annual Conference on Neural Information Processing Systems (NIPS'06)*, pages 985–992. MIT Press, 2007. 2

[23] S. Nowozin, C. Rother, S. Bagon, T. Sharp, B. Yao, and P. Kohli. Decision tree fields. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1668–1675. IEEE, 2011. 2

[24] Q. Qiu and G. Sapiro. Learning transformations for classification forests. *arXiv preprint arXiv:1312.5604*, 2013. 2

[25] J. R. Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986. 1, 4

[26] K. Ricanek and T. Tesafaye. Morph: A longitudinal image database of normal adult age-progression. In *Automatic Face and Gesture Recognition, 2006. FGR 2006. 7th International Conference on*, pages 341–345. IEEE, 2006. 6

[27] M. Robnik-Šikonja. Improving random forests. In *Machine Learning: European Conference on Machine Learning (ECML) 2004*, pages 359–370. Springer, 2004. 2

[28] S. Schulter, C. Leistner, P. Wohlhart, P. M. Roth, and H. Bischof. Alternating regression forests for object detection and pose estimation. In *Proc. International Conference on Computer Vision*, 2013. 2, 4, 5

[29] S. Schulter, P. Wohlhart, C. Leistner, A. Saffari, P. M. Roth, and H. Bischof. Alternating decision forests. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 2, 4, 5, 7

[30] R. Shaoqing, C. Xudong, W. Yichen, and S. Jian. Face alignment at 3000 fps via regressing local binary features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2014. 1

[31] J. Shotton, A. Fitzgibbon, C. Mat, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, 2011. 1, 5

[32] V. Vapnik. *The nature of statistical learning theory*. springer, 2000. 3

[33] A. Verikas, A. Gelzinis, and M. Bacauskiene. Mining data with random forests: A survey and results of new tests. *Pattern Recognition*, 44(2):330–349, 2011. 1