Random Decision DAG: An Entropy Based Compression Approach for Random Forest

Xin Liu[1], Xiao Liu[1], Yongxuan Lai[2], Fan Yang[1,2(✉)], and Yifeng Zeng[3]

[1] Department of Automation, Xiamen University, Xiamen, China
xinliu@stu.xmu.edu.cn, xiaoliu95@outlook.com, yang@xmu.edu.cn
[2] Shenzhen Research Institute/Software School, Xiamen University, Xiamen, China
laiyx@xmu.edu.cn
[3] School of Computing, Teesside University, Middlesbrough, UK
yifeng.zeng.uk@gmail.com

Abstract. Tree ensembles, such as Random Forest (RF), are popular methods in machine learning because of their efficiency and superior performance. However, they always grow big trees and large forests, which limits their use in many memory constrained applications. In this paper, we propose Random decision Directed Acyclic Graph (RDAG), which employs an entropy-based pre-pruning and node merging strategy to reduce the number of nodes in random forest. Empirical results show that the resulting model, which is a DAG, dramatically reduces the model size while achieving competitive classification performance when compared to RF.

Keywords: Random Forest · Pre-pruning · Directed Acyclic Graph

1 Introduction

Tree ensembles, such as Random Forest (RF) [3], are very popular methods in machine learning in the merits of their computational efficiency and overall good performances. However, in order to deal with high-dimensional data, RFs and other tree ensembles often grow big decision trees and large forests, which may not only deteriorate the generalization performances but limit their use in memory-constrained devices, such as mobile phones [2,6]. In this case, post-pruning methods [4], which need to build and store the whole forest in memory first, is not applicable. To build a lightweight model, pre-pruning methods begin to arouse the interest of researchers. For example, Globally Induced Forests (GIFs) [2] iteratively and greedily deepens multiple trees by optimizing a global function and pre-pruning the undesirable nodes. However, GIF uses

Supported by the Natural Science Foundation of China (61672441, 61673324), the Natural Science Foundation of Fujian (2018J01097), the Shenzhen Basic Research Program (JCYJ20170818141325209).

G. Li et al. (Eds.): DASFAA 2019, LNCS 11448, pp. 319–323, 2019.
https://doi.org/10.1007/978-3-030-18590-9_37

extremely randomized trees to generate nodes, and cannot be directly applied to standard random forests. In this paper, we propose Random decision Directed Acyclic Graph (RDAG), which has the following properties:

- It can be viewed as a pre-pruning method. Similar to [2], it avoids generating a whole forest first, therefore, requiring much fewer memories.
- It is fast with a linear time complexity in the number of training instances.
- It results in a DAG with multiple roots while RF results in many redundant trees.

2 RDAG: A Compression Approach for RF

The motivation of RDAG is that the overfitting risk of the tree increases with the node splitting. Pre-pruning can reduce the risk by preventing node splitting, while node merging can also reduce the risk by reducing the number of nodes. The implementation of node splitting and merging is guided by an Adaptive Information Criterion $AdIC$. In a standard RF, the trees are fully grown and independent of each other, while in RDAG the node splitting is guided by $AdIC$ and node merging is performed across different branches in every iteration. Hence we obtain a directed acyclic graph rather than an ensemble of trees. The framework is shown in Algorithm 1. Given a training set with D observations, the algorithm will generate T root nodes with Bagging (Step 2) and then the

Algorithm 1. The Framework of RDAG

Input: $\{x_n, y_n\}_{n=1}^{N}$: training data
T: tree number
λ: parameter of regularization term
n_iter: max depth
Output: F: RDAG model
1: $gNodes \leftarrow \{\}$: growing nodes
2: Grow T root nodes and distribute sampled training data to them (Bagging)
3: $F \leftarrow \{root_1, root_2, \cdots, root_T\}$
4: **for** $i\_iter = 1$ to n_iter **do**
5: update $gNodes$
6: **if** $gNodes$ is empty **then**
7: **break**
8: **end if**
9: split and merge nodes
10: update F
11: **end for**

depth of the graph will be increased by iteration. In every iteration, the node set $gNodes$ including those nodes which should be modified is updated (Step 5), and splitting or merging trail is executed on every node in $gNodes$ (Step 9), then the RDAG model F will be updated (Step 10). Once the node set $gNodes$ is empty, the iteration will be terminated before the graph achieves the maximum depth n_iter (Step 6–7). The overall time complexity of RDAG framework is $O(N \times p \times d)$, which is the same as that of RF, with N, p, d denoting the number of instances, features, and nodes in RDAG respectively.

3 Experimental Results

We investigate RDAG on 20 UCI datasets [1], and a summary of these datasets are listed in Table 1. The baseline method, RF, is performed with version 0.19.1 of Scikit-Learn [5]. The reported results are averaged over 10 times of 10-fold CV. In each fold, an internal 10-fold CV is run on the training partition for parameter tuning.

Table 1. Details of datasets used in experiments

Name	Case	Feature	Category	Name	Case	Feature	Category
hayes	132	4	3	ccc	30000	23	2
car	1728	6	4	dermatology	366	34	6
wholesale	440	7	2	hearts	268	44	3
pima	768	8	2	lung-cancer	32	56	3
breast-cancer	699	9	2	bankrupt1	7027	64	2
cmc	1473	9	3	urban	675	147	9
yeast	1485	9	10	musk2	6598	166	2
heart	303	13	5	arrhythmia	452	261	13
crx	684	15	2	colon	62	2000	2
hepatitis	155	19	2	leukemia	72	5147	2

For RF, the number of trees is set to 10 (denoted as RF_{10}), 100 (denoted as RF_{100}) and 1000 (denoted as RF_{1000}) respectively. For node splitting, the parameter $mtry$ which denoted the number of selected features per splitting is set to $\sqrt{p}$ with p denoting the number of features [3]. For RDAG, it takes the same manner and parameter setting of node splitting as RF. We try $T \in [1, 100]$. Other hyperparameters are tuned by cross-validation.

Table 2. Comparison of test accuracy (in percentage) and the average number of nodes of RF and RDAG: the ordering of the accuracy is marked by different gray values for a clearer comparison. The data is displayed in the format of "average accuracy (standard deviation)—number of nodes".

dataset	RF_{10}	RF_{100}	RF_{1000}	RDAG
hayes	81.21 (1.06)—511	80.97 (1.47)—5030	80.91 (1.30)—49921	81.15 (1.91)—166
car	97.21 (0.20)—3204	98.36 (0.16)—32496	98.64 (0.14)—322563	98.48 (0.13)—2679
wholesale	91.20 (0.48)—644	91.77 (0.32)—6390	91.57 (0.36)—64082	91.66 (0.55)—258
pima	73.56 (0.56)—2444	75.99 (0.54)—23794	76.76 (0.44)—236792	76.78 (1.09)—3126
breast-cancer	95.85 (0.38)—564	96.77 (0.21)—5651	96.92 (0.13)—56292	96.81 (0.14)—632
cmc	51.37 (1.12)—9839	51.94 (0.55)—98342	52.17 (0.61)—983361	53.41 (0.45)—3611
heart	56.59 (1.66)—1569	57.28 (0.80)—15678	57.18 (0.61)—156644	57.89 (1.22)—152
crx	86.39 (0.59)—1812	88.08 (0.38)—17884	88.16 (0.37)—178410	87.80 (0.67)—2307
hepatitis	84.82 (2.40)—373	84.58 (1.33)—3748	85.10 (1.51)—37514	85.68 (1.15)—191
ccc	80.55 (0.78)—78219	81.56 (0.50)—782345	81.63 (0.51)—7821372	81.78 (0.80)—158
dermatology	96.80 (0.68)—624	97.26 (0.34)—6244	97.51 (0.19)—62455	97.40 (0.53)—276
hearts	79.68 (1.36)—504	81.16 (1.18)—5015	81.30 (0.68)—50473	81.68 (1.59)—601
lung-cancer	43.50 (6.79)—184	46.67 (3.78)—1850	44.67 (4.09)—18632	47.67 (4.16)—114
bankrupt1	96.98 (0.61)—3431	97.64 (0.37)—32947	97.67 (0.40)—332848	97.74 (0.35)—1462
urban	82.99 (1.02)—1399	86.05 (0.40)—13902	86.19 (0.31)—138402	85.05 (0.67)—1407
musk2	97.47 (0.49)—3238	97.91 (0.49)—32488	98.00 (0.43)—325225	98.12 (0.47)—4521
arrhythmia	69.60 (1.32)—1584	73.90 (0.94)—15855	74.01 (0.50)—158601	74.16 (1.02)—1772
colon	75.24 (4.05)—109	82.00 (2.89)—1071	81.79 (2.22)—10722	80.88 (2.75)—136
leukemia	91.80 (1.69)—77	96.21 (1.42)—777	97.62 (0.92)—7785	95.84 (0.86)—144

The test accuracy and model size are shown in Table 2. As the number of trees in RF increases, the overall performances are getting better, while the node size also increases dramatically. RDAG achieves similar accuracy with RF_{1000}, while it has much fewer nodes. Even compared to RF_{10}, the number of nodes in RDAG is always smaller. Figure 1 shows a comparison on node distribution on *hearts*. It shows the node distribution at different depths so that we can see the effect of compression. The nodes of RF increase exponentially with the depth unless the instances are all allocated to leaf nodes, and a large number of leaf nodes are generated at a relatively shallow depth, while the sample sizes of these nodes are very sparse. In contrast, RDAG limits the growing width of the model and necessarily seeks compensation in depth to obtain sufficient learning ability.

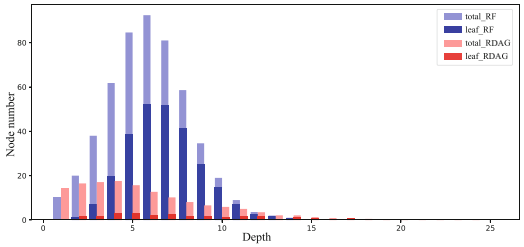

Fig. 1. Comparison on node distribution of RF_{10} and RDAG on *hearts*.

4 Conclusions

In this paper, we propose an entropy-based compression approach for random forests which generates a lightweight classification model. Experimental results show that the resulted RDAG model is compact and accurate.

References

1. Bache, K., Lichman, M.: UCI machine learning repository (2013). http://archive.ics.uci.edu/ml
2. Begon, J.M., Joly, A., Geurts, P.: Globally induced forest: a prepruning compression scheme. In: International Conference on Machine Learning, pp. 420–428 (2017)
3. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
4. Elisha, O., Dekel, S.: Wavelet decompositions of random forests: smoothness analysis, sparse approximation and applications. J. Mach. Learn. Res. **17**(1), 6952–6989 (2016)
5. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**(Oct), 2825–2830 (2011)
6. Shotton, J., Sharp, T., Kohli, P., Nowozin, S., Winn, J., Criminisi, A.: Decision jungles: compact and rich models for classification. In: Advances in Neural Information Processing Systems, pp. 234–242 (2013)