Relative Forest for Attribute Prediction

Shaoxin Li[1,2], Shiguang Shan[1], Xilin Chen[1]

[1]Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, Beijing, 100190, China
[2]Graduate University of Chinese Academy of Sciences, Beijing 100049, China
{shaoxin.li, shiguang.shan, xilin.chen}@vipl.ict.ac.cn

Abstract. Human-Namable visual attributes are promising in leveraging various recognition tasks. Intuitively, the more accurate the attribute prediction is, the more the recognition tasks can benefit. Relative attributes [1] learns a ranking function per attribute which can provide more accurate attribute prediction, thus, show clear advantages over previous binary attribute. In this paper, we inherit the idea of learning ranking function per attribute but propose to improve the algorithm in two aspects: First, we propose a *Relative Tree* algorithm which facilitates more accurate nonlinear ranking to capture the semantic relationships. Second, we develop a *Relative Forest* algorithm which resorts to randomized learning to reduce training time of *Relative Tree*. Benefiting from multiple tree ensemble, *Relative Forest* can achieve even more accurate final ranking. To show the effectiveness of proposed method, we first compare *Relative Tree* method with Relative Attribute on PubFig and OSR dataset. Then to verify the efficiency of *Relative Forest* algorithm, we conduct age estimation evaluation on FG-NET dataset. With much less training time compared to Relative Attribute and *Relative Tree*, proposed *Relative Forest* achieves state-of-the-art age estimation accuracy. Finally, experiments on the large scale SUN Attribute database show the scalability of proposed *Relative Forest*.

1 Introduction

Recently, computer vision researchers have proposed to explore human-namable visual attribute as a valuable semantic cue to boost the performance of traditional recognition tasks [2–5] or enable various new applications [6–10]. While visual attribute shows encouraging capacity in enhancing the robustness of real-world face [2], human [3] and object [4, 5] recognition, the ever growing interest in attribute-centric modeling is even more popularized by its intuitive appeal to facilitate nature image describing [6–8] and knowledge transferring [9, 10].

Notwithstanding the great potential of attribute-centric modeling, few works are devoted to generate informative attribute predictions. Most of existing work [2–10] modeled attribute as binary property and predicted its presence. Although binary prediction is adequate for some attributes, such as having horn and wearing glasses, it is too restrictive and unnatural for a large variety of other attributes, such as human age and object size. As indicated in [1], describing using relative visual properties is a much more informative and effective way for humans to recognize specific objects. Even for properties which are commonly regarded as binary, relative description can provide more information about the image. For example, it is better to describe a scene of country road as "more manmade than a scene of mountains but more natural than tall building" than simply categorize it as natural or manmade. To model such relationship of a given attribute between different classes, Relative Attribute learns a ranking function for each attribute. By estimating a continuous ranking score rather than a binary presence, Relative Attribute can provide much more informative description to enhance subsequence recognition or other tasks. However, since only a linear hyperplane is learned as ranking function, Relative Attribute may not be capable in handling high dimensional nonlinear data.

In this paper, we focus on efficiently generating informative attribute predictions. Inheriting the elegant idea of learning ranking function per attribute with ordered or similar constraints, we propose to improve Relative Attribute in two aspects: 1) Facilitates more accurate nonlinear ranking to cope with high dimensional visual features lying on nonlinear manifold; 2) Reduces time complexity to handle large scale data set. To achieve the first goal, we employ Relative Attribute as base ranking function to construct a Relative Tree according to maximal information gain criterion [11]. Due to the hierarchical tree structure, the proposed Relative Tree can efficiently capture the complex nonlinear structure of feature manifold and generate a piecewise linear ranking function to rank the nonlinear data accurately. Although only linear ranking function is employed, our method can automatically discover the intrinsic nonlinear structure of the data manifold and adapt to arbitrary data distributions. To accomplish the second goal, which is essential for scaling the algorithm up to handle more realistic dataset such as [12] and [13], we first resort to randomized learning to significantly reduce the complexity of building single *Relative Tree*. Then borrowing the idea of random forest [14], we combine multiple randomized *Relative Tree* to construct a *Relative Forest* which can further boosts the ranking accuracy with much less training time.

2 Brief Review of Relative Attribute

Before presenting proposed method, we briefly review Relative Attribute [1]. Unlike most existing attribute-based methods, Relative Attribute learns a ranking function rather than binary prediction per attribute. The key idea is to learn a linear ranking direction to maximize rank margin between given pairs of examples.

Given a set of training samples $\boldsymbol{X} = \{x_k | k = 1, 2, \dots, K\}$, where K is the number of training samples. For each attribute, an ordered constraint set $\boldsymbol{O} = \{o_n | o_n = (k_{n1}, k_{n2}), n = 1, 2, ..., N\}$ and an similar constraint set $\boldsymbol{S} = \{s_m | s_m = (k_{m1}, k_{m2}), m = 1, 2, ..., M\}$ are provided to depict the relationships between pairs of samples. Where N is the number of ordered constraints and M is the number of similar constraints. Each ordered constraint $o_n \in O$ indicates that

$k_{n1} \succ k_{n2}$, i.e. sample k_{n1} has a higher presence of given attribute than k_{n2}, while each similar constraint $s_n \in S$ indicates that $k_{m1} \sim k_{m2}$, i.e. sample k_{m1} has similar attribute presence with sample k_{m2}. Intuitively, the optimal linear ranking direction w should satisfy the ordered and similar constraints as many as possible. As this problem is NP-hard, non-negative slack variables ξ_n, γ_m proposed in [15] are introduced to relax the problem to the final objective of Relative Attribute:

$$\begin{aligned}
\arg\min_{w} \quad & (\frac{1}{2} \parallel w \parallel_2^2 + C(\sum_{n=1}^{N} \xi_n^2 + \sum_{m=1}^{M} \gamma_m^2)) \\
s.t. \quad & w^T(x_{k_{n1}} - x_{k_{n2}}) \geq 1 - \xi_n;\ \forall o_n \in O \\
& |w^T(x_{k_{m1}} - x_{k_{m2}})| \leq \gamma_m;\ \forall s_m \in S \\
& \xi_n, \gamma_m \geq 0,
\end{aligned} \tag{1}$$

where C is used to trade off between large rank margin and number of satisfied constraints. By considering all margins between closest pairs of ordered samples, the learned linear function w is much more suitable for ranking the data than linear classification hyperplane learned from nearest binary-labeled samples. It is important to note that the relative constraint used in Relative Attribute is also more natural to depict some attribute which are hard to be quantized, such as "smile face" and "open scene".

3 Relative Tree

In this section, we propose to extend linear ranking algorithm, i.e. Relative Attribute, to deal with nonlinear ranking. As introduced in [16], a set of tree structured projections can hierarchically partition data into pieces in a manner that is provably sensitive to low dimensional manifold structure. Inspired by this idea, in section 3.1, we propose a method to facilitate hierarchical nonlinear ranking by constructing a *Relative Tree*, in which Relative Attribute serves as base ranking function(also referred to as splitting function) in each tree node. Then in section 3.2 we present how to predict attribute with the constructed *Relative Tree.*

3.1 Tree Construction—Hierarchical Ranking

The basic idea of *Relative Tree* is to learn a set of hierarchical ranking functions. Then by traversing down the tree, a test sample can obtain gradually finer ranking score. Although it is also possible to apply kernel trick to equation (1) for nonlinear ranking, our method bears certain intuitive appeal that the data-driven tree construction involves non hypothesis about data distribution. And all leaf nodes actually consist a global piecewise linear ranking on origin data manifold. Thus our proposed *Relative Tree* can cope with arbitrary data distribution and automatically learns a set of piecewise linear ranking functions

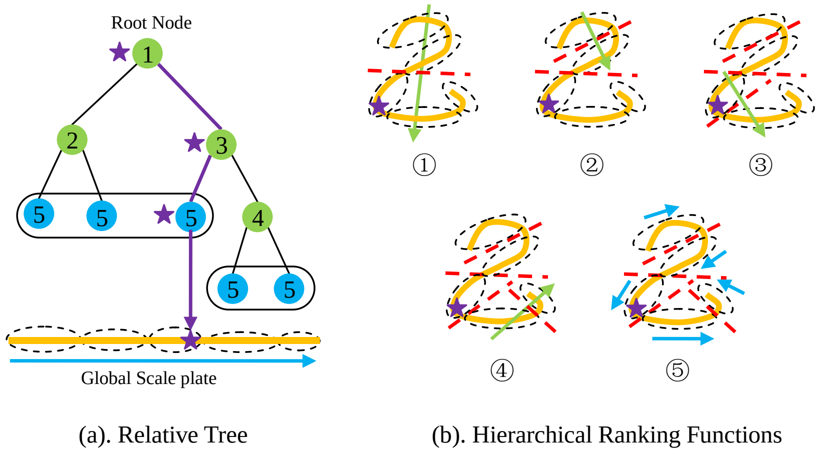

Fig. 1. Learning procedure of Relative Tree(a) on a "S-shape" nonlinear manifold and its corresponding set of Hierarchical Ranking Functions(b). Numbers in the circle indicate train step index. Green color indicates non-leaf nodes while blue color indicates leaf nodes. Ranking functions of all leaf nodes can be normalized to a global "scale-plate" and generate final unified ranking score.

according to the intrinsic structure. We show a sample of construction procedure of proposed *Relative Tree* on a "S-shape" manifold in Fig. 1, where Fig. 1(a) is the learned hierarchical *Relative Tree* and Fig. 1(b) shows the learning steps. The "S-shape" nonlinear manifold can be quantized into 5 approximate linear part. Each of them falls in a dotted elliptical outline in Fig. 1(b). The ranking function learned in the root node(See Fig.1(a)) is actually the result of Relative Attribute algorithm [1]. As is shown, this ranking function(see Fig.1(b): ①) gives almost highest ranking score to the test sample indicated by the purple star. Unlike Relative Attribute, besides of the global coarse ranking function, *Relative Tree* automatically learns finer local ranking functions according to the intrinsic structure of data. As a result, it can predict accurate ranking position of the purple star localizing in the end of the third dotted elliptical outline.

For further clarification, we present the pseudo code of the tree construction procedure in Algorithm. 1, in which there are two key problems remain to be solved: 1) Whether the chosen node is divisible(Algorithm 1 Line: 1); 2) How to find best split(Algorithm Line: 4).

Divisibility. If the presence of attribute is quantifiable, the ordered label can be assigned to samples. In this circumstance, if all the data fall in a node have the same label then this node is strictly indivisible. However, strictly indivisible is almost impossible. Therefore, in our algorithm, we determine a node as indivisible if the entropy of the data falling in this node is less than a given threshold. Once the threshold is small enough the performance will be satisfiable. On the other hand, if only some relative pairs of attribute presence degree are provided, we determine a node as strictly indivisible when there is no ordered

Algorithm 1 : Relative Tree Construction

```
Input:
    Training features: $X$;
    Ordered constraint set: $O$;
    Similar constraint set: $S$.
Output:
    Relative Tree: $T$.
1: if size($X$)>Minimal Node size and $X$ is divisible then
2:     Calculate Relative Attribute: $W$ = RelativeAttribute($X, O, S$);
3:     Calculate ranking score: $R = W^T X$;
4:     Find best splitting threshold $b$;
5:     T.Data = $(R, W, b)$;
6:     Split training data: $X^{(L)} = \{x_k | R_k < b\},\ X^{(R)} = \{x_k | R_k > b\}$;
7:     Split ordered constraint set:
         $O^{(L)} = \{o_n | x_{k_{n1}}, x_{k_{n2}} \in X^{(L)}\};\ O^{(R)} = \{o_n | x_{k_{n1}}, x_{k_{n2}} \in X^{(R)}\}$;
8:     Split similar constraint set:
         $S^{(L)} = \{s_m | x_{k_{m1}}, x_{k_{m2}} \in X^{(L)}\};\ S^{(R)} = \{s_m | x_{k_{m1}}, x_{k_{m2}} \in X^{(R)}\}$;
9:     Save left node $(T.LeftNode, X^{(L)}, O^{(L)}, S^{(L)})$ as a splitting candidate $P$;
10:    Save right node $(T.RightNode, X^{(R)}, O^{(R)}, S^{(R)})$ as a splitting candidate $P$;
11:    Find best splitting node (T,X,O,S) in pool $P$;
12:    Repeat steps 1–15;
13: else
14:    $T = NULL$;
15: end if
16: return $T$;
```

pairs between training samples fall in this node. In real world case, when the ratio of ordered samples which is at least relate to another sample under ordered constraint exceeds a given threshold, we determine this node as divisible.

Split criterion. Once the node is determined as divisible, we need to find the best split to partition the data. To achieve this goal, we first employ Relative Attribute algorithm [1] to learn a linear ranking direction. If ordered label is available, similar to [11], we calculate optimal splitting threshold b to split the data based on maximal information gain criterion:

$$\underset{b}{argmin}\, \Delta E(b) = E(X) - \frac{|X^{(L)}|}{|X|} E(X^{(L)}) - \frac{|X^{(R)}|}{|X|} E(X^{(R)}), \tag{2}$$

where $E(X)$ is the Shannon entropy of the classes in the set of samples X. $X^{(L)}/X^{(R)}$ is the sample data partitioned to the Left/Right child node. If only partial relative pairwise constraints are available, we choose optimal splitting threshold b which separates maximal number of ordered pair and maintains maximal number of similar pair:

$$\underset{b}{argmin}\, \Delta I(b) = I(O, S) - \frac{|X^{(L)}|}{|X|} I(O^{(L)}, S^{(L)}) - \frac{|X^{(R)}|}{|X|} I(O^{(R)}, S^{(R)}), \tag{3}$$

where $I(O,S) = |O| - |S|$, $|O|$ is the number of ordered constraint on X and $|S|$ is the number of similar constraint on $|X|$. Intuitively, the more the ordered pairs and less similar pairs, the more uncertain the data is. In other words, $I(O,S) \propto E(X)$. It is true that more complex and accurate approximation of $E(X)$ can be made with O and S. In this paper, we simply use the heuristic approximation given out by equation (3).

At last, we analyze time complexity of proposed *Relative Tree*. As mentioned in the last paragraph of section 2, the time complexity of Relative Attribute is $O(K^3)$. For proposed *Relative Tree*, in the best case, i.e. we get a complete binary tree and at each node the data is equally distributed to left and right nodes, the complexity is $\sum_{i=0}^{\log_2^K} 2^i O((\frac{K}{2^i})^3) = O(K^3)$. Indeed, if the training data do not severely biased or ordered constraint are uniformly distribute on samples, our method tends to choose balanced split. In our experiments, consuming time of constructing a *Relative Tree* is approximately 3.5 times of training a Relative Attribute in average.

3.2 Tree Prediction—Ranking Score Normalization

With the learned *Relative Tree*, a test sample get gradually finer ranking by traversing down the tree(see Fig. 1(a)). Unlike classical decision tree, which averaging(regression) or voting(classification) training data's labels falling in leaf nodes to generate predictions of test samples, the relative tree will calculate ranking score of the test sample using the leaf node's ranking function. However, the ranking score calculated in different leaf nodes(see Fig. 1) is incomparable since these functions are trained with disjoint subsets of original training set, thus we need to normalize the ranking results obtained by different ranking functions in a *Relative Tree*. The basic idea is to normalize all the ranking score with a unique global "scale plate". If the ordered label is available, the unique global "scale plate" can be directly set as the corresponding label of the data, then the normalization parameter of given ranking function can be obtained by:

$$\arg\min_{s,b} \sum \|l_i - s \cdot (r_i - b)\|_2^2, \tag{4}$$

where l is the ordered label vector and r is the ranking score vector obtained in training step(see Algorithm 1 Line: 3). s and b are normalization parameters which are used to map local rank value to global "scale plate". For the situation when only partial ordered pairwise constraints are given, we can set the unique global "scale plate" as $[0,1]$. We show detailed normalization in Algorithm 2.

After score normalization, the proposed *Relative Tree* algorithm can generate a global ranking score, thus facilitate adaptive piecewise linear ranking on nonlinear data.

4 Relative Forest

Although *Relative Tree* facilitate more accurate nonlinear ranking, it may suffer from over-fitting and is time consuming. In this section, we propose *Relative*

Algorithm 2 : Relative Tree Prediction without Ordered Label

```
Input:
    Relative Tree: $T$;
    Test sample: $x$.
Output:
    Ranking score of give sample: $R_x$.
 1: $b_{inf} = 0;\ b_{sup} = 1$
 2: while T!=NULL do
 3:     $(R, W, b) = T.Data$
 4:     Calculate ranking score of test sample: $r_x = W^T x$
 5:     Normalize training set ranking score to global unique ”scale plate”:
          $R \Rightarrow [b_{inf}, b_{sup}]$, accordingly $b = Norm\_b,\ r_x = Norm\_r_x$
 6:     if $r_x < b$ then
 7:         $T = T.LeftNode;\ b_{sup} = Norm\_b$
 8:     else
 9:         $T = T.RightNode;\ b_{inf} = Norm\_b$
10:     end if
11: end while
12: $R_x = Norm\_r_x$
13: return $R_x$;
```

Forest algorithm based on *Relative Tree*. Resorting to randomized and ensemble learning, compared to *Relative Tree*, the proposed *Relative Forest* obtain even more accurate prediction of attribute while consuming significantly less training time.

4.1 Randomized Learning

In original random forest algorithm [14] and its extending works [17, 18, 11] Randomness usually injected at three aspects during training: 1) bootstrap subset of training data to grow each random relative tree; 2) choose subset of feature dimensions for calculating splitting function f; 3) use random splitting threshold b. In the proposed *Relative Forest*, we employ first two randomization strategies but select a optimal splitting threshold b based on equation (2) or (3).

Additionally, we develop a specific randomized learning strategy for Relative Attribute which serve as node splitting function. As mentioned in the last paragraph of section 2, the total number of relative constraints is $O(K^2)$. When K is large, the number of pairwise constraints becomes untractable. We propose to randomly pick a subset of constraints for optimizing equation (1). Intuitively, we want every sample at least related to another sample with one order constraint and one similar constraint. Suppose that there are c different ordered levels for a given attribute, and in each level n samples are given. Thus totally $K = cn$ samples are provided. In this case, there are totally $|O| = \frac{c(c-1)}{2} n^2$ ordered constrains, in which totally $|O_i| = (c-1)n$ ordered constraints are related to a given sample i. Then by randomly selecting mK ordered constraints, it is easy to see that when $K \gg 1$, the probability that none ordered constraints include sample i is selected is $[1 - \frac{|O_i|}{|O|}]^{mK} \approx e^{-2m}$. For similar constraints, the same criterion also holds. Thus in our experiments, we set $m = 2$, in other words, only randomly choose $2K$ ordered and similar constraints respectively. And the corresponding probability for each sample to be selected in at least one ordered (or similar) constraint is $1 - e^{-4} = 0.9817$. As only $O(K)$ comparing to original $O(K^2)$ constraints are used, when K is very large this strategy can significantly reduce the number of used constraints. These randomized learning strategies not only speed up the training procedure but also reduce over-fitting [19].

4.2 Tree Ensemble

With randomized learning strategies we can build a *Relative Tree* much more faster. However, the ranking accuracy will degrade to some extent. To further boost the performance, we average the ranking score obtained from different randomized *Relative Tree* to generate the final prediction As all the rank scores are normalized to a unique global "scale plate". The average step can be directly applied. Due to the randomization, ranking results of different trees are independent. Thus the ensemble of different random *Relative Tree* improves the estimation accuracy notably as the tree number increase.

5 Experiments

We conduct comprehensive experiments on four datasets: (1) Outdoor Scene Recognition (OSR) Dataset [20] containing 2688 images from 8 categories; (2) A subset of the Public Figure Face (PubFig) Database [2] containing 800 images from 8 random identities (100 images each); (3) FG-NET Face Aging (FG-NET) Dataset [21] containing 1002 images form 82 individuals; (4) Sun Attribute Dataset [12]. For OSR and PubFig dataset, we use exactly the same attributes and data introduced in [1] with exactly the same feature and training\test distribution. For FG-NET, we simply use age as ordered label and conduct leave-one-out experiments according to the evaluation protocol, and for Sun Attribute database We use all 87 attributes collecting in 'asymmetric' splits, but reorganize the train/test split to demonstrate the capability of *Relative Forest* in large scale scenario. For each attribute, we use all image data which receive at least 2 votes as the positive data and randomly select data which receive 0/1 vote as the negative data. We set $N_0 = N_1 = N_2 + N_3$ (N_0, N_1, N_2 and N_3 are the numbers of selected samples receiving 0, 1, 2, 3 vote(s) respectively). Then 5 fold cross validation is constructed for evaluation.

5.1 Relative Tree vs. Relative Attribute

We use the data provided by authors of [1] to conduct experiments in this subsection. For an image pair (i, j) in a test set (2648 images for OSR, 560 for PubFig), we compare ranking scores obtained by different methods namely B-SVM(binary SVM), SVR, RA(Relative Attribute) and RT(Relative Tree). If $R(x_i) > R(x_j)$

Table 1. Relative Accuracy on OSR Dataset(%).

	Natural	Open	Perspective	Large	Diagonal	Close
B-SVM	91.10	86.06	78.98	65.01	80.77	87.20
SVR	94.01	90.42	85.51	86.17	86.87	87.34
RA [1]	94.40	91.01	85.08	86.39	87.52	88.71
RT	95.24	92.39	87.58	88.34	89.43	89.54

Table 2. Relative Accuracy on PubFig Dataset(%).

	Masculine	White	Young	Smiling	Chubby	Forhead
B-SVM	70.12	64.64	75.49	66.97	59.37	76.50
SVR	75.36	69.98	76.25	76.58	72.34	85.79
RA [1]	81.00	77.31	81.05	79.66	76.14	87.91
RT	85.33	82.59	84.41	83.36	78.97	88.83
	Eyebrow	Eye	Nose	Lips	Face	
B-SVM	69.05	74.90	66.29	74.52	74.25	
SVR	75.22	77.08	69.14	71.86	74.23	
RA [1]	78.89	80.72	74.84	78.07	80.46	
RT	81.84	83.15	80.43	81.87	86.31	

we predict $i \succ j$ else $i \prec j$, then this prediction are compared to the ground-truth relative ordering. The accuracy of predictions is shown in Table 1 (OSR) and Table 2 (PubFig). As seen, the proposed *Relative Tree* outperform all competitive algorithms in all given attributes.

As mentioned in section 3.2, once ordered label is given, we regard it as unique global "scale plate" and normalize the ranking score according to it. In this circumstance, the normalized ranking score of test sample can be directly compared with given ordered label of each individual using Euclidean distance in order to recognize it. With this simple method, recognition accuracy using attribute prediction of *Relative Tree* is 84.37% and 70.24% on the OSR and PubFig datasets, respectively, as compared to 77.40% and 65.54% if using ranking score obtained by Relative Attribute.

5.2 Relative Forest vs. Relative Tree

We compare *Relative Forest* to *Relative Tree* in two aspects: Prediction Accuracy and Time Consumption. The evaluation is conducted on the FG-NET data set. Images of the first person (15 images) in this data set are used as the test set, and images of all the other person (987 images) are used for training. Mean Absolute Error(MAE) is used to evaluate the estimation accuracy. The evaluation results are shown in Fig. 2. The dotted green line is the results of Relative Attribute with the score normalization using equation (4). The dotted blue line is the results of *Relative Tree* and the solid red line is the results of *Relative Forest*. In *Relative Forest*, we randomly bootstrap 40% samples with replacement for each tree. In tree construction step, 30% features are randomly selected to train

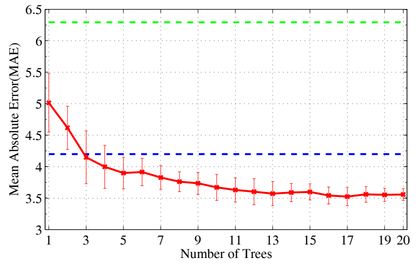

Fig. 2. Age estimation result of *Relative Forest* as the tree number increases.

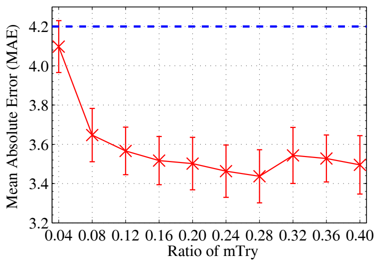

Fig. 3. Analysis of selected feature number of *Relative Forest*.

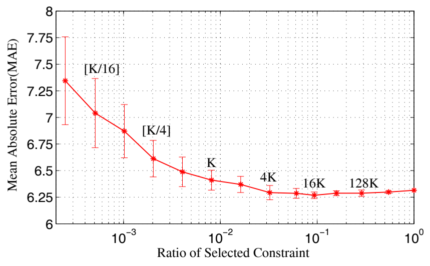

Fig. 4. Analysis of selected constraints number of Relative Attributes [1].

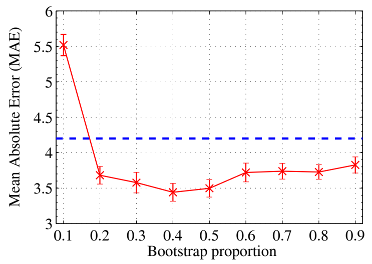

Fig. 5. Analysis of bootstrap sample number of *Relative Forest*

splitting function, i.e. Relative Attribute. We try 10 times and select the best based on information gain criterion [11]. When the size of forest grows larger than 5, *Relative Forest* performs consistently better than *Relative Tree*. For each method, we repeat the algorithm 10 times and calculate mean consumption time. The consumption times are 121.8s and 417.7s for Relative Attribute and *Relative Tree* respectively. When the number of tree is 20, *Relative Forest* consumes 60.3s.

We also analyze effect of three kinds of randomization introduced in subsection 4.1. The corresponding key parameters are: 1)number of bootstrap samples for training each single tree (see Fig. 5); 2)selected dimensions for training Relative Attribute in node splitting step (see Fig. 3); 3)selected number of constraints used in training Relative Attribute (see Fig. 4). Note that the default parameters of tree number, bootstrap proportion, mTry Ratio and constraint number are 20, 40%, 30% and $2K$ respectively. When analyzing one parameter, we fix the other three.

5.3 Relative Forest for Age Estimation

We conduct facial age estimation on FG-NET dataset. Similar to [22], we first use OLPP [23] to extract informative feature from gray scale face image, then use proposed *Relative Forest* to predict age of subjects. The MAE results of our method and state-of-the-art age estimation methods [24–26, 22] are shown

Table 3. MAEs Comparison on FG-NET aging database.

Methods	OHRanker [24]	MTWGP [25]	MHR [26]	LARR [22]	SVR [27]	**Relative Forest**
MAEs	4.48	4.83	4.87	5.07	5.79	**4.45±0.02**

in Table 3. With the same features used in [22], our method achieve the best MAE result on FG-NET database. With the same feature we used, MAE result of Support Vector Regression(SVR) is also shown in Table 3 for more clear comparison.

5.4 Relative Forest for Large Scale Attribute Prediction

We also present results on the SUN attribute database [12]. GIST feature is used. All three comparison methods predict continuous label. We set an attribute as present if predicted label is larger than 0.5. We calculate all 87*5 prediction accuracies, and the mean accuracies are: $Acc_{SVR} = 67.78; Acc_{RA} = 68.29; Acc_{RF} = 69.57;$. Note that for Relative Attribute method we simply sample part of the data and part of positive and negative constraints to train ranking function, or it cannot scale to such large scale testing.

6 Conclusion

Human-Namable visual attribute bears intuitive appeal in strengthening object recognition, facilitating zero shot learning, enabling knowledge transfer and image describing, which make the accurate estimation of attribute in large scale scenario especially important. Aiming at efficiently predicting accurate relative attribute, we first propose *Relative Tree* to facilitate more precise nonlinear ranking, and then propose *Relative Forest* to further boost the performance while significantly reducing training time.

7 Acknowledgements

This work is partially supported by Natural Science Foundation of China (NSFC) under contract Nos. 61025010, 61173065, and 60833013; and Beijing Natural Science Foundation (New Technologies and Methods in Intelligent Video Surveillance for Public Security) under contract No. 4111003.

References

1. Parikh, D., Grauman, K.: Relative attributes. In: ICCV. (2011) 503–510
2. Kumar, N., Berg, A., Belhumeur, P., Nayar, S.: Attribute and simile classifiers for face verification. In: ICCV. (2009) 365–372

3. Yao, B., Jiang, X., Khosla, A., Lin, A., Guibas, L., Fei-Fei, L.: Human action recognition by learning bases of action attributes and parts. In: ICCV. (2011) 1331–1338
4. Wang, J., Markert, K., Everingham, M.: Learning models for object recognition from natural language descriptions. In: BMVC. (2009)
5. Wang, G., Forsyth, D.: Joint learning of visual attributes, object classes and visual saliency. In: ICCV. (2009) 537–544
6. Farhadi, A., Endres, I., Hoiem, D., Forsyth, D.: Describing objects by their attributes. In: CVPR. (2009) 1778–1785
7. Bourdev, L., Maji, S., Malik, J.: Describing people: A poselet-based approach to attribute classification. In: ICCV. (2011) 1543–1550
8. Siddiquie, B., Feris, R., Davis, L.: Image ranking and retrieval based on multi-attribute queries. In: CVPR. (2011) 801–808
9. Fergus, R., Bernal, H., Weiss, Y., Torralba, A.: Semantic label sharing for learning with many categories. In: ECCV. (2010) 762–775
10. Lampert, C., Nickisch, H., Harmeling, S.: Learning to detect unseen object classes by between-class attribute transfer. In: CVPR. (2009) 951–958
11. Shotton, J., Johnson, M., Cipolla, R.: Semantic texton forests for image categorization and segmentation. In: CVPR. (2008) 1–8
12. Patterson, G., Hays, J.: Sun attribute database:discovering, annotating, and recognizing scene attributes. In: CVPR. (2012) 2751–2758
13. Xiao, J., Hays, J., Ehinger, K., Oliva, A., Torralba, A.: Sun database: Large-scale scene recognition from abbey to zoo. In: CVPR. (2010) 3485–3492
14. Breiman, L.: Random forests. Machine learning (2001) 5–32
15. Joachims, T.: Optimizing search engines using clickthrough data. In: SIGKDD. (2002) 133–142
16. Freund, Y., Dasgupta, S., Kabra, M., Verma, N.: Learning the structure of manifolds using random projections. In: NIPS. (2007) 473–480
17. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Machine learning (2006) 3–42
18. Bosch, A., Zisserman, A., Muoz, X.: Image classification using random forests and ferns. In: ICCV. (2007) 1–8
19. Amit, Y., Geman, D.: Shape quantization and recognition with randomized trees. Neural computation (1997) 1545–1588
20. Oliva, A., Torralba, A.: Modeling the shape of the scene: A holistic representation of the spatial envelope. IJCV (2001) 145–175
21. FGNET: The fg-net aging database. http://sting.cycollege.ac.cy/ alanitis/fgnetaging/index.htm (2002)
22. Guo, G., Fu, Y., Dyer, C., Huang, T.: Image-based human age estimation by manifold learning and locally adjusted robust regression. TIP (2008) 1178–1188
23. Cai, D., He, X., Han, J., Zhang, H.: Orthogonal laplacianfaces for face recognition. TIP (2006) 3608–3614
24. Chang, K., Chen, C., Hung, Y.: Ordinal hyperplanes ranker with cost sensitivities for age estimation. In: CVPR. (2011) 585–592
25. Zhang, Y., Yeung, D.: Multi-task warped gaussian process for personalized age estimation. In: CVPR. (2010) 2622–2629
26. Qin, T., Zhang, X., Wang, D., Liu, T., Lai, W., Li, H.: Ranking with multiple hyperplanes. In: SIGIR. (2007) 279–286
27. Chang, C., Lin, C.: Libsvm: a library for support vector machines. TIST (2011) 27