Deep Regression Forests for Age Estimation

Wei Shen[1,2], Yilu Guo[1], Yan Wang[2], Kai Zhao[3], Bo Wang[4], Alan Yuille[2]
[1] Key Laboratory of Specialty Fiber Optics and Optical Access Networks,
Shanghai Institute for Advanced Communication and Data Science,
School of Communication and Information Engineering, Shanghai University
[2] Department of Computer Science, Johns Hopkins University
[3] College of Computer and Control Engineering, Nankai University [4] Hikvision Research
{shenwei1231,gyl.luan0,wyanny.9,zhaok1206,wangbo.yunze,alan.l.yuille}@gmail.com

Abstract

Age estimation from facial images is typically cast as a nonlinear regression problem. The main challenge of this problem is the facial feature space w.r.t. ages is inhomogeneous, due to the large variation in facial appearance across different persons of the same age and the non-stationary property of aging patterns. In this paper, we propose Deep Regression Forests (DRFs), an end-to-end model, for age estimation. DRFs connect the split nodes to a fully connected layer of a convolutional neural network (CNN) and deal with inhomogeneous data by jointly learning input-dependant data partitions at the split nodes and data abstractions at the leaf nodes. This joint learning follows an alternating strategy: First, by fixing the leaf nodes, the split nodes as well as the CNN parameters are optimized by Back-propagation; Then, by fixing the split nodes, the leaf nodes are optimized by iterating a step-size free update rule derived from Variational Bounding. We verify the proposed DRFs on three standard age estimation benchmarks and achieve state-of-the-art results on all of them.

1. Introduction

There has been a growing interest in age estimation from facial images, driven by the increasing demands for a variety of potential applications in forensic research [2], security control [24], human-computer interaction (HCI) [24] and social media [46]. Although this problem has been extensively studied, the ability to automatically estimate ages accurately and reliably from facial images is still far from meeting human performance.

There are two kinds of age estimation tasks. One is real age estimation, which is to estimate the precise biological (chronological) age of a person from his or her facial image; the other is age group estimation [37], which is to predict whether a person's age falls within some range rather than predicting the real chronological age. In this paper, we focus on the first task, i.e., precise age regression. To address this problem, the key is to learn a nonlinear mapping function between facial image features and the real chronological age. However, to learn such a mapping is challenging. The main difficulty is the facial feature space w.r.t. ages is inhomogeneous, due to two facts: 1) there is a large variation in facial appearance across different persons of the same age (Fig. 1(a)); 2) the human face matures in different ways at different ages, e.g., bone growth in childhood and skin wrinkles in adulthood [43] (Fig. 1(b)).

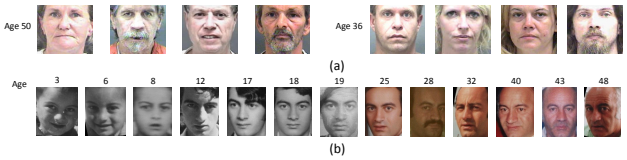

Figure 1. (a) The large variation in facial appearance across different persons of the same age. (b) Facial images of a person from childhood to adulthood. Note that, Facial aging effects appear as changes in the shape of the face during childhood and changes in skin texture during adulthood, respectively.

To model such inhomogeneous data, existing age estimation methods either find a kernel-based global nonlinear mapping [23, 20], or apply divide-and-conquer strategies to partition the data space and learn multiple local regressors [25]. However, each of them has drawbacks: Learning non-stationary kernels is inevitably biased by the inhomogeneous data distribution and thus easily causes over-fitting [5]; Divide-and-conquer is a good strategy to learn the non-stationary age changes in human faces, but the existing methods make hard partitions according to ages [26, 25]. Consequently, they may not find homogeneous subsets for learning local regressors [29].

To address the above-mentioned challenges, we propose differentiable regression forests for age estimation. Random forests or randomized decision trees [3, 4, 12], are a popular ensemble predictive model, in which each tree

structure naturally performs data partition at split nodes and data abstraction at leaf nodes [49]. Traditional regression forests make hard data partitions, based on heuristics such as using a greedy algorithm where locally-optimal hard decisions are made at each split node [3]. Unlike them, the proposed differentiable regression forests perform soft data partition, so that an input-dependent partition function can be learned to handle inhomogeneous data. In addition, the input feature space and the data abstractions at leaf nodes (local regressors) can be learned jointly, which ensures that the local input-output correlation is homogeneous at the leaf node.

Recently, end-to-end learning with CNN has become very popular and has been shown to be useful for improving the performance of various computer vision tasks, such as image classification [35], semantic segmentation [38] and object detection [44, 13]. Our differentiable regression forests can be seamlessly integrated with any deep networks, which enables us to conduct an end-to-end deep age estimation model, named by Deep Regression Forests (DRFs). To build such a tree based model, we apply an alternating optimization strategy: first we fix the leaf nodes and optimize the data partitions at split nodes as well as the CNN parameters (feature learning) by Back-propagation; Then, we fix the split nodes and optimize the data abstractions at leaf nodes (local regressors) by Variational Bounding [33, 57]. These two learning steps are alternatively performed to jointly optimize feature learning and regression modeling for age estimation.

We evaluate our algorithm on three standard benchmarks for real age estimation methods: MORPH [45], FG-NET [42] and the Cross-Age Celebrity Dataset (CACD) [8]. Experimental results demonstrate that our algorithm outperforms several state-of-the-art methods on these three benchmarks.

Our algorithm was inspired by Deep Neural Decision Forests (dNDFs) [34] and Label Distribution Learning Forests (LDLFs) [50], but differs in its objective (regression *vs* classification/label distribution learning). Extending differentiable decision trees to deal with regression is non-trivial, as the distribution of the output space for regression is continuous, but the distribution of the output space for the two classification tasks is discrete. The contribution of this paper is three folds:

1) We propose Deep Regression Forests (DRFs), an end-to-end model, to deal with inhomogeneous data by jointly learning input-dependant data partition at split nodes and data abstraction at leaf nodes.

2) Based on Variational Bounding, the convergence of our update rule for leaf nodes in DRFs is mathematically guaranteed.

3) We apply DRFs on three standard age estimation benchmarks, and achieve state-of-the-art results.

2. Related Work

Age Estimation One way to tackle precise facial age estimation is to search for a kernel-based global non-linear mapping, like kernel support vector regression [23] or kernel partial least squares regression [20]. The basic idea is to learn a low-dimensional embedding of the aging manifold [19]. However, global non-linear mapping algorithms may be biased [29], due to the heterogenous properties of the input data. Another way is to adopt divide-and-conquer approaches, which partition the data space and learn multiple local regressors. But hierarchical regression [25] or tree based regression [40] approaches made hard partitions according to ages, which is problematic because the subsets of facial images may not be homogeneous for learning local regressors. Huang *et al.* [29] proposed Soft-margin Mixture of Regressions (SMMR) to address this issue, which found homogenous partitions in the joint input-output space, and learned a local regressor for each partition. But their regression model cannot be integrated with any deep networks as an end-to-end model.

Several researchers formulated age estimation as an ordinal regression problem [7, 41, 10], because the relative order among the age labels is also important information. They trained a series of binary classifiers to partition the samples according to ages, and estimated ages by summing over the classifier outputs. Thus, ordinal regression is limited by its lack of scalability [29]. Some other researchers formulated age estimation as a label distribution learning (LDL) problem [15], which paid attention to modeling the cross-age correlations, based on the observation that faces at close ages look similar. LDL based age estimation methods [16, 17, 55, 50] achieved promising results, but the label distribution model is usually inflexible in adapting to complex face data domains with diverse cross-age correlations [27].

With the rapid development of deep networks, more and more end-to-end CNN based age estimation methods [46, 41, 1] have been proposed to address this non-linear regression problem. But how to deal with inhomogeneous data is still an open issue.

Random Forests Random forests are an ensemble of randomized decision trees [4]. Each decision tree consists of several split nodes and leaf nodes. Tree growing is usually based on greedy algorithms which make locally-optimal hard data partition decisions at each split node. Thus, this makes it intractable to integrate decision trees and deep networks in an end-to-end learning manner. Some effort has been made to combine these two worlds [34, 31, 36]. The newly proposed Deep Neural Decision Forests (dNDFs) [34] addressed this shortcoming by defining a soft partition function at each split node, which enabled the decision trees to be differentiable, allowing joint learning with deep networks. Shen *et al.* [50] then extended this differ-

entiable decision tree to address label distribution learning problems. As we mentioned in Sec. 1, our DRFs model is inspired by these two works, but differs in the objective (regression *vs* classification/label distribution learning). One recent work proposed Neural Regression Forest (NRF) [48] for depth estimation, which is similar to our DRFs. But mathematically, the convergence of their update rule for leaf nodes was not guaranteed.

3. Deep Regression Forests

In this section, we first introduce how to learn a single differentiable regression tree, then describe how to learn tree ensembles to form a forest.

3.1. Problem Formulation

Let $\mathcal{X} = \mathbb{R}^{d_x}$ and $\mathcal{Y} = \mathbb{R}^{d_y}$ denote the input and output spaces, respectively. We consider a regression problem, where for each input sample $\mathbf{x} \in \mathcal{X}$, there is an output target $\mathbf{y} \in \mathcal{Y}$. The objective of regression is to find a mapping function $\mathbf{g} : \mathbf{x} \rightarrow \mathbf{y}$ between an input sample $\mathbf{x}$ and its output target $\mathbf{y}$. A standard way to address this problem is to model the conditional probability function $p(\mathbf{y}|\mathbf{x})$, so that the mapping is given by $\hat{\mathbf{y}} = \mathbf{g}(\mathbf{x}) = \int \mathbf{y} p(\mathbf{y}|\mathbf{x}) d\mathbf{y}$.

We propose to model this conditional probability by a decision tree based structure $\mathcal{T}$. A decision regression tree consists of a set of split nodes $\mathcal{N}$ and a set of leaf nodes $\mathcal{L}$. Each split node $n \in \mathcal{N}$ defines a split function $s_n(\cdot; \mathbf{\Theta}) : \mathcal{X} \rightarrow [0, 1]$ parameterized by $\mathbf{\Theta}$ to determine whether a sample is sent to the left or right subtree. Each leaf node $\ell \in \mathcal{L}$ contains a probability density distribution $\pi_\ell(\mathbf{y})$ over $\mathcal{Y}$, i.e, $\int \pi_\ell(\mathbf{y}) d\mathbf{y} = 1$. Following [34, 50], we use a soft split function $s_n(\mathbf{x}; \mathbf{\Theta}) = \sigma(f_{\varphi(n)}(\mathbf{x}; \mathbf{\Theta}))$, where $\sigma(\cdot)$ is a sigmoid function, $\varphi(\cdot)$ is an index function to bring the $\varphi(n)$-th output of function $\mathbf{f}(\mathbf{x}; \mathbf{\Theta})$ in correspondence with a split node n, and $\mathbf{f} : \mathbf{x} \rightarrow \mathbb{R}^M$ is a real-valued feature learning function depending on the sample $\mathbf{x}$ and the parameter $\mathbf{\Theta}$. $\mathbf{f}$ can take any forms. In our DRFs, it is a CNN and $\mathbf{\Theta}$ is the network parameter. The index function $\varphi(\cdot)$ specifies the correspondence between the split nodes and output units of $\mathbf{f}$ (it is initialized randomly before tree learning). An example to demonstrate the sketch chart of our DRFs as well as $\varphi(\cdot)$ is shown in Fig. 2 (There are two trees with index functions, φ_1 and φ_2 for each). Then, the probability of the sample $\mathbf{x}$ falling into leaf node ℓ is given by

$$P(\ell|\mathbf{x}; \mathbf{\Theta}) = \prod_{n \in \mathcal{N}} s_n(\mathbf{x}; \mathbf{\Theta})^{\mathbf{1}(\ell \in \mathcal{L}_{n_l})} (1 - s_n(\mathbf{x}; \mathbf{\Theta}))^{\mathbf{1}(\ell \in \mathcal{L}_{n_r})}, \tag{1}$$

where $\mathbf{1}(\cdot)$ is an indicator function and $\mathcal{L}_{n_l}$ and $\mathcal{L}_{n_r}$ denote the sets of leaf nodes held by the subtrees $\mathcal{T}_{n_l}$, $\mathcal{T}_{n_r}$ rooted at the left and right children n_l, n_r of node n (shown in Fig. 3), respectively. The conditional probability function

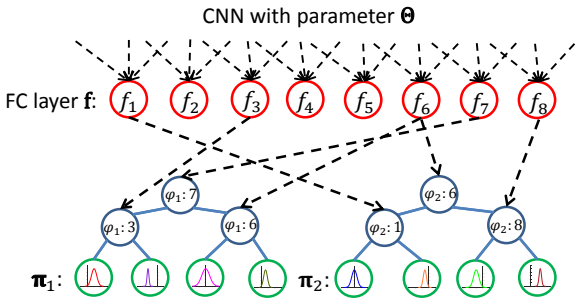

Figure 2. Illustration of a deep regression forest (DRF). The top red circles denote the output units of the function $\mathbf{f}$ parameterized by $\mathbf{\Theta}$. Here, they are the units of a fully-connected (FC) layer in a CNN. The blue and green circles are split nodes and leaf nodes, respectively. Two index functions φ_1 and φ_2 are assigned to these two trees respectively. The black dash arrows indicate the correspondence between the split nodes of these two trees and the output units of the FC layer. Note that, one output unit may correspond to the split nodes belonging to different trees. Each tree has independent leaf node distribution $\boldsymbol{\pi}$ (denoted by distribution curves in leaf nodes). The output of the forest is a mixture of the tree predictions. $\mathbf{f}(\cdot; \mathbf{\Theta})$ and $\boldsymbol{\pi}$ are learned jointly in an end-to-end manner.

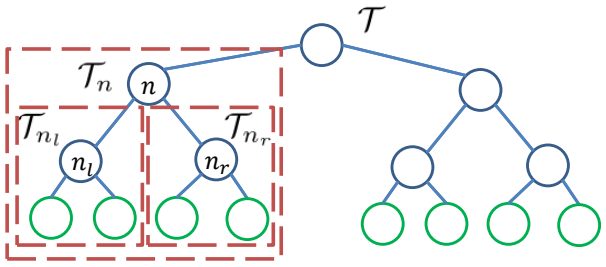

Figure 3. The subtree rooted at node n: $\mathcal{T}_n$ and its left and right subtrees: $\mathcal{T}_{n_l}$ and $\mathcal{T}_{n_r}$.

$p(\mathbf{y}|\mathbf{x}; \mathcal{T})$ given by the tree $\mathcal{T}$ is

$$p(\mathbf{y}|\mathbf{x}; \mathcal{T}) = \sum_{\ell \in \mathcal{L}} P(\ell|\mathbf{x}; \mathbf{\Theta}) \pi_\ell(\mathbf{y}). \tag{2}$$

Then the mapping between $\mathbf{x}$ and $\mathbf{y}$ modeled by tree $\mathcal{T}$ is given by $\hat{\mathbf{y}} = \mathbf{g}(\mathbf{x}; \mathcal{T}) = \int \mathbf{y} p(\mathbf{y}|\mathbf{x}; \mathcal{T}) d\mathbf{y}$.

3.2. Tree Optimization

Given a training set $\mathcal{S} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$, learning a regression tree $\mathcal{T}$ described in Sec. 3.1 leads to minimizing the following negative log likelihood loss:

$$\begin{aligned} R(\boldsymbol{\pi}, \mathbf{\Theta}; \mathcal{S}) &= -\frac{1}{N} \sum_{i=1}^{N} \log(p(\mathbf{y}_i|\mathbf{x}_i, \mathcal{T})) \\ &= -\frac{1}{N} \sum_{i=1}^{N} \log\Big(\sum_{\ell \in \mathcal{L}} P(\ell|\mathbf{x}_i; \mathbf{\Theta}) \pi_\ell(\mathbf{y}_i)\Big), \end{aligned} \tag{3}$$

where $\boldsymbol{\pi}$ denotes the density distributions contained by all the leaf nodes $\mathcal{L}$. Note that, optimizing $R(\boldsymbol{\pi}, \boldsymbol{\Theta}; \mathcal{S})$ requires estimating both the split node parameter $\boldsymbol{\Theta}$ and the density distributions $\boldsymbol{\pi}$ held by leaf nodes, i.e.,

$$(\boldsymbol{\Theta}^*, \boldsymbol{\pi}^*) = \arg\min_{\boldsymbol{\Theta}, \boldsymbol{\pi}} R(\boldsymbol{\pi}, \boldsymbol{\Theta}; \mathcal{S}). \tag{4}$$

To solve Eqn. 4, we alternate the following two steps: (1) fixing $\boldsymbol{\pi}$ and optimizing $\boldsymbol{\Theta}$; (2) fixing $\boldsymbol{\Theta}$ and optimizing $\boldsymbol{\pi}$, until convergence or a maximum number of iterations is reached.

3.2.1 Learning Split Nodes by Gradient Descent

Now, we discuss how to learn the parameter $\boldsymbol{\Theta}$ for split nodes, when the density distributions held by the leaf nodes $\boldsymbol{\pi}$ are fixed. Thanks to the soft split function, the tree loss $R(\boldsymbol{\pi}, \boldsymbol{\Theta}; \mathcal{S})$ is differentiable with respect to $\boldsymbol{\Theta}$. The gradient of the loss is computed by the chain rules as follows:

$$\frac{\partial R(\boldsymbol{\pi}, \boldsymbol{\Theta}; \mathcal{S})}{\partial \boldsymbol{\Theta}} = \sum_{i=1}^{N} \sum_{n \in \mathcal{N}} \frac{\partial R(\boldsymbol{\pi}, \boldsymbol{\Theta}; \mathcal{S})}{\partial f_{\varphi(n)}(\mathbf{x}_i; \boldsymbol{\Theta})} \frac{\partial f_{\varphi(n)}(\mathbf{x}_i; \boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}. \tag{5}$$

Note that in the right part of Eqn. 5, only the first term depends on the tree and the second term depends only on the specific form of the function $f_{\varphi(n)}$. The first term is computed by

$$\frac{\partial R(\boldsymbol{\pi}, \boldsymbol{\Theta}; \mathcal{S})}{\partial f_{\varphi(n)}(\mathbf{x}_i; \boldsymbol{\Theta})} = \frac{1}{N} \Big(s_n(\mathbf{x}_i; \boldsymbol{\Theta}) \Gamma^i_{n_r} - \big(1 - s_n(\mathbf{x}_i; \boldsymbol{\Theta})\big) \Gamma^i_{n_l} \Big), \tag{6}$$

where for a generic node $n \in \mathcal{N}$

$$\Gamma^i_n = \frac{p(\mathbf{y}_i | \mathbf{x}_i; \mathcal{T}_n)}{p(\mathbf{y}_i | \mathbf{x}_i; \mathcal{T})} = \frac{\sum_{\ell \in \mathcal{L}_n} P(\ell | \mathbf{x}_i; \boldsymbol{\Theta}) \pi_\ell(\mathbf{y}_i)}{p(\mathbf{y}_i | \mathbf{x}_i; \mathcal{T})}. \tag{7}$$

Γ^i_n can be efficiently computed for all nodes n in the tree $\mathcal{T}$ by a single pass over the tree. Observing that $\Gamma^i_n = \Gamma^i_{n_l} + \Gamma^i_{n_r}$, the computation for Γ^i_n can be started at the leaf nodes and conducted in a bottom-up manner. Based on Eqn. 6, the split node parameters $\boldsymbol{\Theta}$ can be learned by standard Back-propagation.

3.2.2 Learning Leaf Nodes by Variational Bounding

By fixing the split node parameters $\boldsymbol{\Theta}$, Eqn. 4 becomes a constrained optimization problem:

$$\min_{\boldsymbol{\pi}} R(\boldsymbol{\pi}, \boldsymbol{\Theta}; \mathcal{S}), \textbf{s.t.}, \forall \ell, \int \pi_\ell(\mathbf{y}) d\mathbf{y} = 1. \tag{8}$$

For efficient computation, we represent each density distribution $\pi_\ell(\mathbf{y})$ by a parametric model. Since ideally each leaf node corresponds to a compact homogeneous subset, we assume that the density distribution $\pi_\ell(\mathbf{y})$ in each leaf node is a Gaussian distribution, i.e.,

$$\pi_\ell(\mathbf{y}) = \frac{1}{\sqrt{(2\pi)^k \det(\boldsymbol{\Sigma}_\ell)}} \exp\Big(-\frac{1}{2}(\mathbf{y} - \boldsymbol{\mu}_\ell)^{\mathrm{T}} \Sigma_\ell^{-1} (\mathbf{y} - \boldsymbol{\mu}_\ell)\Big), \tag{9}$$

where $\boldsymbol{\mu}_\ell$ and $\boldsymbol{\Sigma}_\ell$ are the mean and the covariance matrix of the Gaussian distribution. Based on this assumption, Eqn. 8 is equivalent to minimizing $R(\boldsymbol{\pi}, \boldsymbol{\Theta}; \mathcal{S})$ w.r.t. $\boldsymbol{\mu}_\ell$ and $\boldsymbol{\Sigma}_\ell$. Now, we propose to address this optimization problem by Variational Bounding [33, 57]. In variational bounding, an original objective function to be minimized gets replaced by a sequence of bounds minimized in an iterative manner. To obtain an upper bound of $R(\boldsymbol{\pi}, \boldsymbol{\Theta}; \mathcal{S})$, we apply Jensen's inequality to it:

$$\begin{aligned}
R(\boldsymbol{\pi}, \boldsymbol{\Theta}; \mathcal{S}) &= -\frac{1}{N} \sum_{i=1}^{N} \log\Big(\sum_{\ell \in \mathcal{L}} P(\ell | \mathbf{x}_i; \boldsymbol{\Theta}) \pi_\ell(\mathbf{y}_i) \Big) \\
&= -\frac{1}{N} \sum_{i=1}^{N} \log\Big(\sum_{\ell \in \mathcal{L}} \zeta_\ell(\bar{\boldsymbol{\pi}}; \mathbf{x}_i, \mathbf{y}_i) \frac{P(\ell | \mathbf{x}_i; \boldsymbol{\Theta}) \pi_\ell(\mathbf{y}_i)}{\zeta_\ell(\bar{\boldsymbol{\pi}}; \mathbf{x}_i, \mathbf{y}_i)} \Big) \\
&\leq -\frac{1}{N} \sum_{i=1}^{N} \sum_{\ell \in \mathcal{L}} \zeta_\ell(\bar{\boldsymbol{\pi}}; \mathbf{x}_i, \mathbf{y}_i) \log\Big(\frac{P(\ell | \mathbf{x}_i; \boldsymbol{\Theta}) \pi_\ell(\mathbf{y}_i)}{\zeta_\ell(\bar{\boldsymbol{\pi}}; \mathbf{x}_i, \mathbf{y}_i)} \Big) \\
&= R(\bar{\boldsymbol{\pi}}, \boldsymbol{\Theta}; \mathcal{S}) - \frac{1}{N} \sum_{i=1}^{N} \sum_{\ell \in \mathcal{L}} \zeta_\ell(\bar{\boldsymbol{\pi}}; \mathbf{x}_i, \mathbf{y}_i) \log\Big(\frac{\pi_\ell(\mathbf{y}_i)}{\bar{\pi}_\ell(\mathbf{y}_i)} \Big),
\end{aligned} \tag{10}$$

where $\zeta_\ell(\boldsymbol{\pi}; \mathbf{x}_i, \mathbf{y}_i) = \frac{P(\ell | \mathbf{x}_i; \boldsymbol{\Theta}) \pi_\ell(\mathbf{y}_i)}{p(\mathbf{y}_i | \mathbf{x}_i; \mathcal{T})}$. Note that $\zeta_\ell(\boldsymbol{\pi}; \mathbf{x}_i, \mathbf{y}_i)$ has the property that $\zeta_\ell(\boldsymbol{\pi}; \mathbf{x}_i, \mathbf{y}_i) \in [0, 1]$ and $\sum_{\ell \in \mathcal{L}} \zeta_\ell(\boldsymbol{\pi}; \mathbf{x}_i, \mathbf{y}_i) = 1$ to ensure that Eqn. 10 holds Jensen's inequality. Let us define

$$\phi(\boldsymbol{\pi}, \bar{\boldsymbol{\pi}}) = R(\bar{\boldsymbol{\pi}}, \boldsymbol{\Theta}; \mathcal{S}) - \frac{1}{N} \sum_{i=1}^{N} \sum_{\ell \in \mathcal{L}} \zeta_\ell(\bar{\boldsymbol{\pi}}; \mathbf{x}_i, \mathbf{y}_i) \log\Big(\frac{\pi_\ell(\mathbf{y}_i)}{\bar{\pi}_\ell(\mathbf{y}_i)} \Big). \tag{11}$$

Then $\phi(\boldsymbol{\pi}, \bar{\boldsymbol{\pi}})$ is an upper bound for $R(\boldsymbol{\pi}, \boldsymbol{\Theta}; \mathcal{S})$, which has the properties that for any $\boldsymbol{\pi}$ and $\bar{\boldsymbol{\pi}}$, $\phi(\boldsymbol{\pi}, \bar{\boldsymbol{\pi}}) \geq \phi(\boldsymbol{\pi}, \boldsymbol{\pi}) = R(\boldsymbol{\pi}, \boldsymbol{\Theta}; \mathcal{S})$ and $\phi(\bar{\boldsymbol{\pi}}, \bar{\boldsymbol{\pi}}) = R(\bar{\boldsymbol{\pi}}, \boldsymbol{\Theta}; \mathcal{S})$. These two properties hold the conditions for Variational Bounding.

Recall that we parameterize $\pi_\ell(\mathbf{y})$ by two parameters: the mean $\boldsymbol{\mu}_\ell$ and the covariance matrix $\boldsymbol{\Sigma}_\ell$. Let $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ denote these two parameters held by all the leaf nodes $\mathcal{L}$. We define $\psi(\boldsymbol{\mu}, \bar{\boldsymbol{\mu}}) = \phi(\boldsymbol{\pi}, \bar{\boldsymbol{\pi}})$, then $\psi(\boldsymbol{\mu}, \bar{\boldsymbol{\mu}}) \geq \phi(\boldsymbol{\pi}, \boldsymbol{\pi}) = \psi(\boldsymbol{\mu}, \boldsymbol{\mu}) = R(\boldsymbol{\pi}, \boldsymbol{\Theta}; \mathcal{S})$, which indicates that $\psi(\boldsymbol{\mu}, \bar{\boldsymbol{\mu}})$ is also an upper bound for $R(\boldsymbol{\pi}, \boldsymbol{\Theta}; \mathcal{S})$. Assume that we are at a point $\boldsymbol{\mu}^{(t)}$ corresponding to the t-th iteration, then $\psi(\boldsymbol{\mu}, \boldsymbol{\mu}^{(t)})$ is an upper bound for $R(\boldsymbol{\mu}, \boldsymbol{\Theta}; \mathcal{S})$. In the next iteration, $\boldsymbol{\mu}^{(t+1)}$ is chosen such that $\psi(\boldsymbol{\mu}^{(t+1)}, \boldsymbol{\mu}) \leq R(\boldsymbol{\mu}^{(t)}, \boldsymbol{\Theta}; \mathcal{S})$, which implies $R(\boldsymbol{\mu}^{(t+1)}, \boldsymbol{\Theta}; \mathcal{S}) \leq R(\boldsymbol{\mu}^{(t)}, \boldsymbol{\Theta}; \mathcal{S})$. Therefore, we can minimize $\psi(\boldsymbol{\mu}, \bar{\boldsymbol{\mu}})$ instead of $R(\boldsymbol{\mu}, \boldsymbol{\Theta}; \mathcal{S})$ after ensuring that

$R(\boldsymbol{\mu}^{(t)}, \boldsymbol{\Theta}; \mathcal{S}) = \psi(\boldsymbol{\mu}^{(t)}, \bar{\boldsymbol{\mu}})$, i.e., $\bar{\boldsymbol{\mu}} = \boldsymbol{\mu}^{(t)}$. Thus, we have

$$\boldsymbol{\mu}^{(t+1)} = \arg\min_{\boldsymbol{\mu}} \psi(\boldsymbol{\mu}, \boldsymbol{\mu}^{(t)}). \tag{12}$$

The partial derivative of $\psi(\boldsymbol{\mu}, \boldsymbol{\mu}^{(t)})$ w.r.t. $\boldsymbol{\mu}_\ell$ is computed by

$$\begin{aligned}
&\frac{\partial \psi(\boldsymbol{\mu}, \boldsymbol{\mu}^{(t)})}{\partial \boldsymbol{\mu}_\ell} = \frac{\partial \phi(\boldsymbol{\pi}, \boldsymbol{\pi}^{(t)})}{\partial \boldsymbol{\mu}_\ell} \\
&= -\frac{1}{N} \sum_{i=1}^{N} \zeta_\ell(\boldsymbol{\pi}^{(t)}; \mathbf{x}_i, \mathbf{y}_i) \frac{\partial \log(\pi_\ell(\mathbf{y}_i))}{\partial \boldsymbol{\mu}_\ell} \\
&= -\frac{1}{N} \sum_{i=1}^{N} \zeta_\ell(\boldsymbol{\pi}^{(t)}; \mathbf{x}_i, \mathbf{y}_i) \boldsymbol{\Sigma}_\ell^{-1} (\mathbf{y}_i - \boldsymbol{\mu}_\ell).
\end{aligned} \tag{13}$$

By setting $\frac{\partial \psi(\boldsymbol{\mu}, \boldsymbol{\mu}^{(t)})}{\partial \boldsymbol{\mu}_\ell} = \mathbf{0}$, where $\mathbf{0}$ denotes zero vector or matrix, we have

$$\boldsymbol{\mu}_\ell^{(t+1)} = \frac{\sum_{i=1}^{N} \zeta_\ell(\boldsymbol{\pi}^{(t)}; \mathbf{x}_i, \mathbf{y}_i) \mathbf{y}_i}{\sum_{i=1}^{N} \zeta_\ell(\boldsymbol{\pi}^{(t)}; \mathbf{x}_i, \mathbf{y}_i)}. \tag{14}$$

Similarly, we define $\xi(\boldsymbol{\Sigma}, \bar{\boldsymbol{\Sigma}}) = \phi(\boldsymbol{\pi}, \bar{\boldsymbol{\pi}})$, then

$$\boldsymbol{\Sigma}^{(t+1)} = \arg\min_{\boldsymbol{\Sigma}} \xi(\boldsymbol{\Sigma}, \boldsymbol{\Sigma}^{(t)}). \tag{15}$$

The partial derivative of $\xi(\boldsymbol{\Sigma}, \boldsymbol{\Sigma}^{(t)})$ w.r.t. $\boldsymbol{\Sigma}_\ell$ is obtained by

$$\begin{aligned}
&\frac{\partial \xi(\boldsymbol{\Sigma}, \boldsymbol{\Sigma}^{(t)})}{\partial \boldsymbol{\Sigma}_\ell} = \frac{\partial \phi(\boldsymbol{\pi}, \boldsymbol{\pi}^{(t)})}{\partial \boldsymbol{\Sigma}_\ell} \\
&= -\frac{1}{N} \sum_{i=1}^{N} \zeta_\ell(\boldsymbol{\pi}^{(t)}; \mathbf{x}_i, \mathbf{y}_i) \frac{\partial \log(\pi_\ell(\mathbf{y}_i))}{\partial \boldsymbol{\Sigma}_\ell} \\
&= -\frac{1}{N} \sum_{i=1}^{N} \zeta_\ell(\boldsymbol{\pi}^{(t)}; \mathbf{x}_i, \mathbf{y}_i) \big[-\frac{1}{2} \boldsymbol{\Sigma}_\ell^{-1} \\
&+ \frac{1}{2} \boldsymbol{\Sigma}_\ell^{-1} (\mathbf{y}_i - \boldsymbol{\mu}_\ell^{(t+1)}) (\mathbf{y}_i - \boldsymbol{\mu}_\ell^{(t+1)})^{\mathrm{T}} \boldsymbol{\Sigma}_\ell^{-1} \big]
\end{aligned} \tag{16}$$

By Setting $\frac{\partial \xi(\boldsymbol{\Sigma}, \boldsymbol{\Sigma}^{(t)})}{\partial \boldsymbol{\Sigma}_\ell} = \mathbf{0}$, we have

$$\sum_{i=1}^{N} \zeta_\ell(\boldsymbol{\pi}^{(t)}; \mathbf{x}_i, \mathbf{y}_i) \big[-\boldsymbol{\Sigma}_\ell + (\mathbf{y}_i - \boldsymbol{\mu}_\ell^{(t+1)}) (\mathbf{y}_i - \boldsymbol{\mu}_\ell^{(t+1)})^{\mathrm{T}} \big] = \mathbf{0}, \tag{17}$$

which leads to

$$\boldsymbol{\Sigma}_\ell^{(t+1)} = \frac{\sum_{i=1}^{N} \zeta_\ell(\boldsymbol{\pi}^{(t)}; \mathbf{x}_i, \mathbf{y}_i) (\mathbf{y}_i - \boldsymbol{\mu}_\ell^{(t+1)}) (\mathbf{y}_i - \boldsymbol{\mu}_\ell^{(t+1)})^{\mathrm{T}}}{\sum_{i=1}^{N} \zeta_\ell(\boldsymbol{\pi}^{(t)}; \mathbf{x}_i, \mathbf{y}_i)}. \tag{18}$$

Eqn. 14 and Eqn. 18 are the update rule for the density distribution $\boldsymbol{\pi}$ held by all leaf nodes, which are step-size free and fast-converged. One issue remained is how to initialize the starting point $\boldsymbol{\mu}_\ell^{(0)}$ and $\boldsymbol{\Sigma}_\ell^{(0)}$. The simplest way is to do k-means clustering on $\{\mathbf{y}_i\}_{i=1}^N$ to obtain $|\mathcal{L}|$ subsets, then initialize $\boldsymbol{\mu}_\ell^{(0)}$ and $\boldsymbol{\Sigma}_\ell^{(0)}$ according to cluster assignment, i.e., let $\mathbb{I}_i$ denote cluster index assigned to $\mathbf{y}_i$, then

$$\begin{aligned}
\boldsymbol{\mu}_\ell^{(0)} &= \frac{\sum_{i=1}^{N} \mathbf{1}(\mathbb{I}_i = \ell) \mathbf{y}_i}{\sum_{i=1}^{N} \mathbf{1}(\mathbb{I}_i = \ell)}, \\
\boldsymbol{\Sigma}_\ell^{(0)} &= \frac{\sum_{i=1}^{N} \mathbf{1}(\mathbb{I}_i = \ell) (\mathbf{y}_i - \boldsymbol{\mu}_\ell^{(0)}) (\mathbf{y}_i - \boldsymbol{\mu}_\ell^{(0)})^{\mathrm{T}}}{\sum_{i=1}^{N} \mathbf{1}(\mathbb{I}_i = \ell)}.
\end{aligned} \tag{19}$$

This initialization can be understood in this way that we first perform data partition only according to ages by k-means, and then the input facial feature space and output age space are jointly learned to find homogeneous partitions during tree building.

3.2.3 Learning a Regression Forest

A regression forest is an ensemble of regression trees $\mathcal{F} = \{\mathcal{T}^1, \ldots, \mathcal{T}^K\}$, where all trees can possibly share the same parameters in $\boldsymbol{\Theta}$, but each tree can have a different set of split functions (assigned by φ, as shown in Fig. 2), and independent leaf node distribution $\boldsymbol{\pi}$. We define the loss function for a forest as the averaged loss functions of all individual trees: $R_{\mathcal{F}} = \frac{1}{K} \sum_{k=1}^{K} R_{\mathcal{T}^k}$, where $R_{\mathcal{T}^k}$ is the loss function for tree $\mathcal{T}^k$ defined by Eqn. 3. Learning the forest $\mathcal{F}$ also follows the alternating optimization strategy described in Sec. 3.2.

Algorithm 1 The training procedure of a DRF.

Require: $\mathcal{S}$: training set, n_B: the number of mini-batches to update $\boldsymbol{\pi}$
  Initialize $\boldsymbol{\Theta}$ randomly and $\boldsymbol{\pi}$ by Eqn. 19. Set $\mathcal{B} = \{\emptyset\}$
  while Not converge **do**
    while $|\mathcal{B}| < n_B$ **do**
      Randomly select a mini-batch B from $\mathcal{S}$
      Update $\boldsymbol{\Theta}$ by computing gradient (Eqn. 20) on B
      $\mathcal{B} = \mathcal{B} \bigcup B$
    end while
    Update $\boldsymbol{\pi}$ by iterating Eqn. 14 and Eqn. 18 on $\mathcal{B}$
    $\mathcal{B} = \{\emptyset\}$
  end while

To learn $\boldsymbol{\Theta}$, by referring to Fig. 2 and our derivation in Sec. 3.2.1, we have

$$\frac{\partial R_{\mathcal{F}}}{\partial \boldsymbol{\Theta}} = \frac{1}{K} \sum_{i=1}^{N} \sum_{k=1}^{K} \sum_{n \in \mathcal{N}_k} \frac{\partial R_{\mathcal{T}^k}}{\partial f_{\varphi_k(n)}(\mathbf{x}_i; \boldsymbol{\Theta})} \frac{\partial f_{\varphi_k(n)}(\mathbf{x}_i; \boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}, \tag{20}$$

where $\mathcal{N}_k$ and $\varphi_k(\cdot)$ are the split node set and the index function of $\mathcal{T}^k$, respectively. The index function $\varphi_k(\cdot)$ for each tree is randomly assigned before tree learning, which means the split nodes of each tree are connected to a randomly selected subset of output units of $\mathbf{f}$. This strategy

is similar to the random subspace method [28], which can increase the randomness in training to reduce the risk of overfitting.

As each tree in the forest $\mathcal{F}$ has its own leaf node distribution $\boldsymbol{\pi}$, we update them independently according to Eqn. 14 and Eqn. 18. In our implementation, we do not conduct this update scheme on the whole dataset $\mathcal{S}$ but on a set of mini-batches $\mathcal{B}$. The training procedure of a DRF is shown in Algorithm. 1.

In the testing stage, the output of the forest $\mathcal{F}$ is given by averaging the predictions from all the individual trees:

$$\begin{aligned}\hat{\mathbf{y}} &= \mathbf{g}(\mathbf{x};\mathcal{F}) = \frac{1}{K}\sum_{k=1}^{K}\mathbf{g}(\mathbf{x};\mathcal{T}^k) \\ &= \frac{1}{K}\sum_{k=1}^{K}\int \mathbf{y}p(\mathbf{y}|\mathbf{x};\mathcal{T}^k)d\mathbf{y} \\ &= \frac{1}{K}\sum_{k=1}^{K}\int \mathbf{y}\sum_{\ell\in\mathcal{L}^k}P(\ell|\mathbf{x};\boldsymbol{\Theta})\pi_\ell(\mathbf{y})d\mathbf{y} \\ &= \frac{1}{K}\sum_{k=1}^{K}\sum_{\ell\in\mathcal{L}^k}P(\ell|\mathbf{x};\boldsymbol{\Theta})\boldsymbol{\mu}_\ell, \end{aligned} \tag{21}$$

where $\mathcal{L}^k$ is the leaf node set of the k-th tree. Here, we take the fact that the expectation of the Gaussian distribution $\pi_\ell(\mathbf{y})$ is $\boldsymbol{\mu}_\ell$.

4. Experiments

In this section we introduce the implementation details and report the performance of the proposed algorithm as well as the comparison to other competitors.

4.1. Implementation Details

Our realization of DRFs is based on the public available "caffe" [32] framework. Following the recent deep learning based age estimation method [46], we use the VGG-16 Net [51] as the CNN part of the proposed DRFs.

Parameters Setting The model-related hyper-parameters (and the default values we used) are: number of trees (5), tree depth (6), number of output units produced by the feature learning function (128), iterations to update leaf-node predictions (20), number of mini-batches used to update leaf node predictions (50). The network training based hyper-parameters (and the values we used) are: initial learning rate (0.05), mini-batch size (16), maximal iterations (30k). We decrease the learning rate ($\times 0.5$) every 10k iterations.

Preprocessing and Data Augmentation Following the previous method [41], faces are firstly detected by using a standard face detectior [52] and facial landmarks are localized by AAM [11]. We perform face alignment to guarantee all eyeballs stay at the same position in the image.

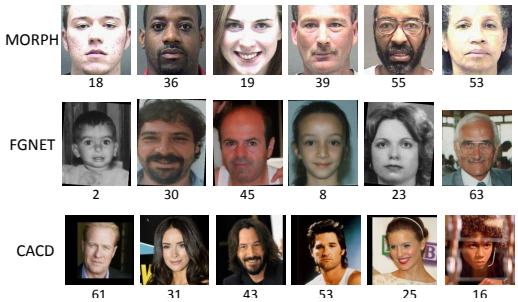

Figure 4. Some examples of MORPH [45], FG-NET [42] and CACD [8]. The number below each image is the chronological age of each subject.

Data augmentation is crucial to train good deep networks. We augment the training data by: (a) cropping images at random offsets, (b) adding gaussian noise to the original images, (c) randomly flipping (left-right).

4.2. Experimental Results

4.2.1 Evaluation Metric

The performance of age estimation is evaluated in terms of mean absolute error (MAE) as well as Cumulative Score (CS). MAE is the average absolute error over the testing set, and the Cumulative Score is calculated by $\text{CS}(l) = \frac{K_l}{K}\cdot 100\%$, where K is the total number of testing images and K_l is the number of testing facial images whose absolute error between the estimated age and the ground truth age is not greater than l years. Here, we set the same error level 5 as in [7, 9, 30], i.e., $l = 5$. Note that, because only some methods reported the Cumulative Score, we are only able to give CS values for some competitors.

4.2.2 Performance Comparison

In this section we compare our DRFs with other state-of-the-art age estimation methods on three standard benchmarks: MORPH [45], FG-NET [42] and the Cross-Age Celebrity Dataset (CACD) [8]. Some examples of these three datasets are illustrated in Fig. 4.

MORPH We first compare DRFs with other state-of-the-art age estimation methods on MORPH, which is the most popular dataset for age estimation. MORPH contains more than 55,000 images from about 13,000 people of different races. Each of the facial image is annotated with a chronological age. The ethnicity of MORPH is very unbalanced, as more than 96% of the facial images are from African or European people.

Existing methods adopted different experimental settings on MORPH. The first setting (Setting I) is introduced in [7, 9, 19, 53, 47, 46, 1], which selects 5,492 images of Caucasian Descent people from the original MORPH dataset, to reduce the cross-ethnicity effects. In Setting I, these 5,492 images are randomly partitioned into two subsets: 80% of the images are selected for training and others

Method	MAE	CS
Human workers [25]	6.30	51.0 %*
AGES [18]	8.83	46.8 %*
MTWGP [58]	6.28	52.1%*
CA-SVR [9]	5.88	57.9%
SVR [19]	5.77	57.1%
OHRank [7]	6.07	56.3%
DLA [53]	4.77	63.4 %*
Rank [6]	6.49	49.1%*
Rothe *et al*. [47]	3.45	N/A
DEX [46]	3.25	N/A
dLDLF [50]	3.02	81.3%
ARN [1]	3.00	N/A
DRFs(ours)	**2.91**	**82.9%**

Table 1. Performance comparison on MORPH [45] (Setting I)(*: the value is read from the reported CS curve).

for testing. The random partition is repeated 5 times, and the final performance is averaged over these 5 different partitions. The second setting is used in [16, 50, 15, 14], under which all of the images in MORPH are randomly split into training/testing (80%/20%) sets. And also the random splitting is performed 5 times repeatedly. The final performance is obtained by averaging the performances of these 5 different splitting. There are also several methods [20, 22, 56] using the third setting (Setting III), which randomly selected a subset (about 21,000 images) from MORPH and restricted the ratio between Black and White and the one between Female and Male are 1:1 and 1:3, respectively. For a fair comparison, we test the proposed DRFs on MORPH under all these three settings. The quantitative results of the three settings are summarized in Table 1, Table 2 and Table 3, respectively. As can be seen from these tables, DRFs achieve the best performance on all of the settings, and outperform the current state-of-the-arts with a clear margin. There is only one method, dLDLF [50], which can achieve slightly worse result than DRFs (for setting II), as this method is also based on differentiable decision forests, but used for label distribution learning.

FG-NET We then conduct experiments on FG-NET [42], a dataset also widely used for age estimation. It contains 1002 facial images of 82 individuals, in which most of them are white people. Each individual in FG-NET has more than 10 photos taken at different ages. The images in FG-NET have a large variation in lighting conditions, poses and expressions.

Following the experimental setting used in [54, 19, 5, 9, 46], we perform "leave one out" cross validation on this dataset, i.e., we leave images of one person for testing and take the remaining images for training. The quantitative comparisons on FG-NET dataset are shown in Table 4. As can be seen, DRFs achieve the state-of-the-art result with 3.85 MAE. Note that, it is the only method that has a MAE below 4.0. The age distribution of FG-NET is strongly biased, moreover, the "leave one out" cross validation policy further aggravates the bias between the training set and the testing set. The ability of overcoming the bias between training and testing sets indicates that the proposed DRFs can handle inhomogeneous data well.

Method	MAE	CS
IIS-LDL [16]	5.67	71.2%*
CPNN [17]	4.87	N/A
Huerta *et al*. [30]	4.25	71.2%
BFGS-LDL [15]	3.94	N/A
OHRank [7]	3.82	N/A
OR-SVM [6]	4.21	68.1%*
CCA [21]	4.73	60.5%*
LSVR [23]	4.31	66.2%*
OR-CNN [41]	3.27	73.0%*
SMMR [29]	3.24	N/A
Ranking-CNN [10]	2.96	85.0%*
DLDL [14]	2.42	N/A
dLDLF [50]	2.24	N/A
DRFs(ours)	**2.17**	**91.3%**

Table 2. Performance comparison on MORPH [45] (Setting II)(*: the value is read from the reported CS curve).

Method	MAE
KPLS [20]	4.18
Guo and Mu [22]	3.92
CPLF [56]	3.63
DRFs(ours)	**2.98**

Table 3. Performance comparison on MORPH [45] (Setting III).

Method	MAE	CS
Human workers [25]	4.70	69.5%*
Rank [6]	5.79	66.5%*
DIF [25]	4.80	74.3%*
AGES [18]	6.77	64.1%*
IIS-LDL [16]	5.77	N/A
CPNN [17]	4.76	N/A
MTWGP [58]	4.83	72.3%*
CA-SVR [9]	4.67	74.5%
LARR [19]	5.07	68.9%*
OHRank [7]	4.48	74.4%
DLA [53]	4.26	N/A
CAM [39]	4.12	73.5%*
Rothe *et al*. [47]	5.01	N/A
DEX [46]	4.63	N/A
DRFs (Ours)	**3.85**	**80.6%**

Table 4. Performance comparison on FG-NET [42](*: the value is read from the reported CS curve).

Trained on	Dex [46]	dLDLF [50]	**DRFs (Ours)**
CACD (train)	4.785	4.734	**4.637**
CACD (val)	6.521	6.769	**5.768**

Table 5. Performance comparison on CACD (measured by MAE) [8].

CACD CACD [8] is a large dataset which has around 160,000 facial images of 2,000 celebrities. These celebrities are divided into three subsets: the training set which is composed of 1,800 celebrities, the testing set that has 120 celebrities and the validation set containing 80 celebrities. Following [46], we evaluate the performance of the models trained on the training set and the validation set, respectively. The detailed comparisons are shown in Table .5. The proposed DRFs model performs better than the competitor DEX [46], no matter which set they are trained on. It's worth noting that, the improvement of DRFs to DEX is much more significant when they are trained on the validation set than the training set. This result can be explained in this way: As we described earlier, the inhomogeneous data is the main challenge in training age estimation models. This challenge can be alleviated by enlarging the scale of training data. Therefore, DEX and our DRFs achieve comparable results when they are trained on the training set. But when they are trained on the validation set, which is much smaller than the training set, DRFs outperform DEX significantly, because we directly address the inhomogeneity challenge. Therefore, DRFs are capable of handling inhomogeneous data even learned from a small set.

4.3. Discussion

4.3.1 Visualization of Learned Leaf Nodes

To better understand DRFs, we visualize the distributions at leaf nodes learned on MORPH [45] (Setting I) in Fig. 5(b). Each leaf node contains a Gaussian distribution (the vertical and horizontal axes represent probability density and age, respectively). For reference, we also display the histogram of data samples (the vertical axis) with respect to age (the horizontal axis). Observed that, the mixture of these Gaussian distributions learned at leaf nodes is very similar to the histogram of data samples, which indicates our DRFs fit the age data well. The age data in MORPH was sampled mostly below age 60, and densely concentrated around 20's and 40's. So the Gaussian distribution centered around 60 has much larger variance than those centered in the interval between 20 and 50, but has smaller probability density. This is because although these learned Gaussian distributions represent homogeneous local partitions, the number of samples is not necessarily uniformly distributed among partitions. Another phenomenon is these Gaussian distributions are heavily overlapped, which accords with the fact that different people with the same age but have quite different facial appearances.

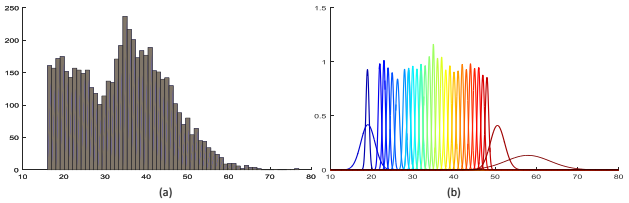

Figure 5. (a) Histogram of data samples with respect to age on MORPH [45] (Setting I). (b) Visualization of the learned leaf node distributions in our DRFs (best viewed in color).

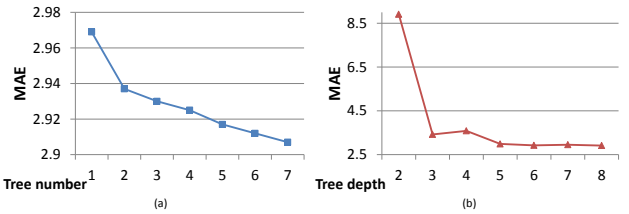

Figure 6. Performance changes by varying (a) tree number and (b) tree depth on MORPH [45] (Setting I).

4.3.2 Parameter Discussion

The tree number and tree depth are two important hyper-parameters for our DRFs. Now we vary each of them and fix the other one to the default value to see how the performance changes on MORPH (Setting I). As shown in Fig. 6, using more trees leads to a better performance as we expected, and with the tree depth increase, the MAE first becomes lower and then stable.

5. Conclusion

We proposed Deep Regression Forests (DRFs) for age estimation, which learn nonlinear regression between inhomogeneous facial feature space and ages. In DRFs, by performing soft data partition at split nodes, the forests can be connected to a deep network and learned in an end-to-end manner, where data partition at split nodes is learned by Back-propagation and data abstraction at leaf nodes is optimized by iterating a step-size free and fast-converged update rule derived from Variational Bounding. The end-to-end learning of split and leaf nodes ensures that partition function at each split node is input-dependent and the local input-output correlation at each leaf node is homogeneous. Experimental results showed that DRFs achieved state-of-the-art results on three age estimation benchmarks.

Acknowledgement. This work was supported in part by the National Natural Science Foundation of China No. 61672336, in part by "Chen Guang" project supported by Shanghai Municipal Education Commission and Shanghai Education Development Foundation No. 15CG43, in part by ONR N00014-15-1-2356 and in part by NSF-MIT NSF CCF-123121.

References

[1] E. Agustsson, R. Timofte, and L. V. Gool. Anchored regression networks applied to age estimation and super resolution. In *Proc. ICCV*, 2017.

[2] K. Alkass, B. A. Buchholz, S. Ohtani, T. Yamamoto, H. Druid, and K. L. Spalding. Age estimation in forensic sciences: Application of combined aspartic acid racemization and radiocarbon analysis. *Mol Cell Proteomics*, 9:1022–1030, 2010.

[3] Y. Amit and D. Geman. Shape quantization and recognition with randomized trees. *Neural Computation*, 9(7):1545–1588, 1997.

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

[5] K. Chang, C. Chen, and Y. Hung. Ordinal hyperplanes ranker with cost sensitivities for age estimation. In *Proc. CVPR*, pages 585–592, 2011.

[6] K.-Y. Chang, C.-S. Chen, and Y.-P. Hung. A ranking approach for human ages estimation based on face images. In *Proc. ICPR*, pages 3396–3399, 2010.

[7] K. Y. Chang, C. S. Chen, and Y. P. Hung. Ordinal hyperplanes ranker with cost sensitivities for age estimation. In *Proc. CVPR*, 2011.

[8] B. Chen, C. Chen, and W. H. Hsu. Face recognition and retrieval using cross-age reference coding with cross-age celebrity dataset. *IEEE Trans. Multimedia*, 17(6):804–815, 2015.

[9] K. Chen, S. Gong, T. Xiang, and C. L. Chen. Cumulative attribute space for age and crowd density estimation. In *Proc. CVPR*, pages 2467–2474, 2013.

[10] S. Chen, C. Zhang, M. Dong, J. Le, and M. Rao. Using ranking-cnn for age estimation. In *Proc. CVPR*, pages 742–751, 2017.

[11] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. In *Proc. ECCV*, pages 484–498, 1998.

[12] A. Criminisi and J. Shotton. *Decision Forests for Computer Vision and Medical Image Analysis*. Springer, 2013.

[13] J. Dai, Y. Li, K. He, and J. Sun. R-FCN: object detection via region-based fully convolutional networks. In *Proc. NIPS*, pages 379–387, 2016.

[14] B. B. Gao, C. Xing, C. W. Xie, J. Wu, and X. Geng. Deep label distribution learning with label ambiguity. *IEEE Transactions on Image Processing*, PP(99):1–1, 2016.

[15] X. Geng. Label distribution learning. *IEEE Transactions on Knowledge and Data Engineering*, 28(7):1734–1748, 2016.

[16] X. Geng, K. Smith-Miles, and Z. Zhou. Facial age estimation by learning from label distributions. In *Proc. AAAI*, 2010.

[17] X. Geng, C. Yin, and Z. Zhou. Facial age estimation by learning from label distributions. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(10):2401–2412, 2013.

[18] X. Geng, Z.-H. Zhou, and K. Smith-Miles. Automatic age estimation based on facial aging patterns. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(12):2234–2240, 2007.

[19] G. Guo, Y. Fu, C. R. Dyer, and T. S. Huang. Image-based human age estimation by manifold learning and locally adjusted robust regression. *IEEE Transactions on Image Processing*, 17(7):1178–1188, 2008.

[20] G. Guo and G. Mu. Simultaneous dimensionality reduction and human age estimation via kernel partial least squares regression. In *Proc. CVPR*, pages 657–664, 2011.

[21] G. Guo and G. Mu. Joint estimation of age, gender and ethnicity: Cca vs. pls. In *Proc. FG*, pages 1–6, 2013.

[22] G. Guo and G. Mu. A framework for joint estimation of age, gender and ethnicity on a large database. *Image and Vision Computing*, 32(10):761–770, 2014.

[23] G. Guo, G. Mu, Y. Fu, and T. S. Huang. Human age estimation using bio-inspired features. In *Proc. CVPR*, pages 112–119, 2009.

[24] H. Han, C. Otto, and A. K. Jain. Age estimation from face images: Human vs. machine performance. In *Proc. ICB*, pages 1–8, 2013.

[25] H. Han, C. Otto, X. Liu, and A. K. Jain. Demographic estimation from face images: Human vs. machine performance. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(6):1148–1161, 2015.

[26] K. Hara and R. Chellappa. Growing regression forests by classification: Applications to object pose estimation. In *Proc. ECCV*, pages 552–567, 2014.

[27] Z. He, X. Li, Z. Zhang, F. Wu, X. Geng, Y. Zhang, M.-H. Yang, and Y. Zhuang. Data-dependent label distribution learning for age estimation. *IEEE Trans. on Image Processing*, 2017.

[28] T. K. Ho. The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(8):832–844, 1998.

[29] D. Huang, L. Han, and F. D. la Torre. Soft-margin mixture of regressions. In *Proc. CVPR*, 2017.

[30] I. Huerta, C. Fernández, and A. Prati. Facial age estimation through the fusion of texture and local appearance descriptors. In *Porc. ECCV Workshops*, pages 667–681, 2014.

[31] Y. Ioannou, D. P. Robertson, D. Zikic, P. Kontschieder, J. Shotton, M. Brown, and A. Criminisi. Decision forests, convolutional networks and the models in-between. *arXiv:1603.01250*, 2016.

[32] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2015.

[33] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999.

[34] P. Kontschieder, M. Fiterau, A. Criminisi, and S. R. Bulò. Deep neural decision forests. In *Proc. ICCV*, pages 1467–1475, 2015.

[35] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. NIPS*, pages 1106–1114, 2012.

[36] C. Lee, P. W. Gallagher, and Z. Tu. Generalizing pooling functions in cnns: Mixed, gated, and tree. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(4):863–875, 2018.

[37] G. Levi and T. Hassner. Age and gender classification using convolutional neural networks. In *Proc. CVPR Workshops*, pages 34–42, 2015.

[38] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proc. CVPR*, pages 3431–3440, 2015.

[39] K. Luu, K. Seshadri, M. Savvides, T. D. Bui, and C. Y. Suen. Contourlet appearance model for facial age estimation. In *Proc. IJCB*, pages 1–8, 2011.

[40] A. Montillo and H. Ling. Age regression from faces using random forests. In *Proc. ICIP*, pages 2465–2468, 2009.

[41] Z. Niu, M. Zhou, L. Wang, X. Gao, and G. Hua. Ordinal regression with multiple output cnn for age estimation. In *Proc. CVPR*, 2016.

[42] G. Panis, A. Lanitis, N. Tsapatsoulis, and T. F. Cootes. Overview of research on facial ageing using the FG-NET ageing database. *IET Biometrics*, 5(2):37–46, 2016.

[43] N. Ramanathan, R. Chellappa, and S. Biswas. Age progression in human faces: A survey. *J. Vis. Lang. Comput.*, 15:3349 – 3361, 2009.

[44] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(6):1137–1149, 2017.

[45] K. Ricanek and T. Tesafaye. MORPH: A longitudinal image database of normal adult age-progression. In *Proc. FG*, pages 341–345, 2006.

[46] R. Rothe, R. Timofte, and L. V. Gool. Deep expectation of real and apparent age from a single image without facial landmarks. *International Journal of Computer Vision*, 2016.

[47] R. Rothe, R. Timofte, and L. V. Gool. Some like it hot - visual guidance for preference prediction. In *Proc. CVPR*, pages 5553–5561, 2016.

[48] A. Roy and S. Todorovic. Monocular depth estimation using neural regression forest. In *Proc. CVPR*, 2016.

[49] W. Shen, K. Deng, X. Bai, T. Leyvand, B. Guo, and Z. Tu. Exemplar-based human action pose correction and tagging. In *Proc. CVPR*, pages 1784–1791, 2012.

[50] W. Shen, K. Zhao, Y. Guo, and A. Yuille. Label distribution learning forests. In *Proc. NIPS*, 2017.

[51] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[52] P. A. Viola and M. J. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. CVPR*, pages 511–518, 2001.

[53] X. Wang, R. Guo, and C. Kambhamettu. Deeply-learned feature for age estimation. In *Proc. WACV*, pages 534–541, 2015.

[54] S. Yan, H. Wang, X. Tang, and T. S. Huang. Learning auto-structured regressor from uncertain nonnegative labels. In *Proc. ICCV*, pages 1–8, 2007.

[55] X. Yang, X. Geng, and D. Zhou. Sparsity conditional energy label distribution learning for age estimation. In *Proc. IJCAI*, pages 2259–2265, 2016.

[56] D. Yi, Z. Lei, and S. Z. Li. Age estimation by multi-scale convolutional network. In *Proc. ACCV*, pages 144–158, 2014.

[57] A. Yuille and A. Rangarajan. The concave-convex procedure. *Neural Computation*, 15(4):915–936, 2003.

[58] Y. Zhang and D.-Y. Yeung. Multi-task warped gaussian process for personalized age estimation. In *Proc. CVPR*, pages 2622–2629, 2010.