RANDOM FOREST ANALYSIS AND ITS APPLICATIONS IN COMPUTER VISION

HAILIANG LI

PhD

The Hong Kong Polytechnic University

2018

THE HONG KONG POLYTECHNIC UNIVERSITY

DEPARTMENT OF ELECTRONIC AND INFORMATION ENGINEERING

Random Forest Analysis and Its Applications in Computer Vision

Hailiang Li

A thesis submitted in partial fulfillment of the requirements

for the degree of Doctor of Philosophy

December 2017

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

__________________________(Signed)

________Hailiang Li_______(Name of student)

Abstract

Random forest is a well-known machine-learning technique, which has been widely employed in machine-learning and computer-vision applications, including classification, clustering and regression, due to its strong generalization power and high efficiency on both its training and inference stages. A random-forest based model consists of an ensemble of binary trees, which build a forest. Every of the trees can make its own decision independently as an individual expert, and the final prediction is the result summarized from all the trees. Although the vanilla version of random forest has been largely used as a standard tool integrated in lots of software packages, random forest is still a research area where some aspects can be improved, such as the criterion in the split nodes, the clustering algorithm in the leaf nodes, and the feature engineering in the whole process.

In this thesis, we address these problems and have proposed some novel ideas which make remarkable improvements on the random-forest-based models. Firstly, we propose a random-forest-based, cascaded regression model for face alignment, by designing a novel locally lightweight feature, namely intimacy definition feature (IDF), which can achieve high speed and stable accuracy for applications where hardware resources are limited, such as mobile devices. Secondly, we present a more accurate face alignment algorithm by combining IDF and the classic constrained local model (CLM) paradigm into a joint framework. After studying the impact of feature distribution in a random forest, we propose a novel random-forest scheme, namely Joint Maximum Purity Forest (JMPF), for classification, clustering, regression and image super-resolution, where the original feature space is transformed into a compactly pre-clustered feature space, via a trained rotation matrix. Finally, we present a novel feature-augmented random forest (FARF) for image super-resolution, where we have studied feature engineering in the whole process of random forest and extended some work from JMPF.

All the algorithms proposed in this thesis have been evaluated and compared to existing state-of-the-art methods. Experimental results and analyses show that our algorithms can achieve stable and promising performances.

List of Publications

[1] **Hailiang Li**, Kin-Man Lam, Dong Li, "Joint maximum purity forest with application to image super-resolution," Journal of Electronic Imaging 27(4), 043005 (2018), doi: 10.1117/1.JEI.27.4.043005.

[2] **Hailiang Li**, Kin-Man Lam, Man-Yau Chiu, Kangheng Wu, Zhibin Lei, "Cascaded face alignment via intimacy definition feature," Journal of Electronic Imaging 26(5), 053024 (2017), doi: 10.1117/1.JEI.26.5.053024.

[3] **Hailiang Li**, Kin-Man Lam, Man-Yau Chiu, Kangheng Wu, and Zhibin Lei. "Efficient Likelihood Bayesian Constrained Local Model.", In 2017 IEEE International Conference on Multimedia and Expo (ICME), (pp. 763-768).

[4] **Hailiang Li**, Kin-Man Lam, Miaohui Wang, "Image Super-Resolution via Feature-Augmented Random Forests", arXiv preprint arXiv:1712.05248 (2017) (on revision for the Journal: Signal Processing: Image Communication).

[5] **Hailiang Li**, Man-Yau Chiu, Kangheng Wu, Zhibin Lei, Kin-Man Lam. "Accurate Car-Plate Detection via Car Face Landmark Localization." The 9th International Conference on Digital Image Processing, (ICDIP 2017).

[6] Fang, Jing, Shuaiqi Liu, Yang Xiao, and **Hailiang Li**. "SAR image de-noising based on texture strength and weighted nuclear norm minimization." Journal of Systems Engineering and Electronics 27, no. 4 (2016): 807-814.

[7] Liu, Shuaiqi, Tao Zhang, **Hailiang Li**, Jie Zhao, and Huiya Li. "Medical image fusion based on nuclear norm minimization." International Journal of Imaging Systems and Technology 25, no. 4 (2015): 310-316.

[8] Yang Liu, Man Yau Chiu, **Hailiang Li**, Kang Heng Wu, Zhibin Lei, Qiang Ding,"A Real Time Robust Hand Tracking Method with Normal Cameras.", Computer Vision: CCF Chinese Conference, CCCV 2015, Xi'an, China

[9] Hu, Jie, **Hailiang Li**, and Ying Li. "Real time super resolution reconstruction for video stream based on GPU." in 2014 IEEE International Conference on Orange Technologies (ICOT), pp. 9-12, 2014.

[10] **Hailiang Li**, and Kin-Man Lam. "Fast super-resolution based on weighted collaborative representation," in IEEE the 19th International Conference on Digital Signal Processing (DSP), pp. 914-918, 2014.

[11] **Hailiang Li**, Tao Cui, Kangheng Wu, and Zhibin Lei. "Two-stage scheme for false alarm suppression on fast face detection." in 2014 International Conference on Anti-counterfeiting, Security and Identification (ASID), pp. 1-5, 2014.

[12] **Hailiang Li**, and Kin-Man Lam. "Guided iterative back-projection scheme for single-image super-resolution." in IEEE Global High-Tech Congress on Electronics (GHTCE), pp. 175-180, 2013.

Acknowledgement

This thesis is the result from not only my hard-working study but also with many people who all supported me in many ways and deserve my deepest gratitude. First and foremost I want to thank my supervisor Prof. Kin-Man Lam for his continuous support of my Ph.D study and research, for his patience, motivation and immense knowledge. Prof. Kin-Man Lam is not only a teacher gave me the change to pursue my PhD at the Department of Electronic and Information Engineering, but also a mentor aroused my interest in the field of computer vision, a friend who understood me when I felt unsure about this PhD journey and encouraged me to get through the tough time.

My sincere thanks also go to Prof. Lei Zhang from the Computing Department, for his help to give me an opportunity to continue my research study in PolyU.

I am grateful to all the DSP lab members I worked with in the past seven years: Muwei Jian, Dong Li, Chensheng Sun, Huiling Zhou, Xiaolong Ma, ChanHong Ho, Jun Xiao, Cigdem Turan and Engr Saad Shakeel Chughtai. I am thankful to their share of knowledge and thoughts at work, and the life we have enjoyed together.

Last but not the least, I would thank my wife and my family for supporting my seven years' study.

Table of Contents

Chapter 1. Introduction

The objective of this chapter is to introduce the general research background of random forest techniques and its applications. Based on the development and some existing problems of the state-of-the-art methods, we discussed our research motivation and the original works proposed in this thesis, along with the outline of the thesis.

1.1 Research Background

The task of machine learning algorithms in computer vision is to learn a function $f(.)$ that maps the representation of the input data to a desired output, which totally depends on the specific task at hand. When we consider high-level computer vision tasks, like image classification or clustering, object detection and tracking, semantic image segmentation, or action recognition, machine learning techniques can be considered as one of its core building blocks to associate a given image with the desired output.

These have been many machine learning techniques successfully employed in computer vision applications to tackle the classification, clustering and regression tasks based on Neural Network (NN), Boosting, Deep Learning, or Support Vector Machine (SVM), etc. In this thesis, however, we focus on Random Forest (RF), a highly flexible and effective learning algorithm that builds on decision trees and has been extensively used in the computer vision community.

1.2 Motivation

I chose random forest as my research topic mainly because random forest works as a traditional ensemble tool, which has the competitive power on big data applications nowadays. Firstly, the most important benefit that makes random forest as a popular choice for the various applications in the machine-learning and computer-vision communities, is the efficiency on both the training and inference stages. The efficiency of random forest comes from its random sampling on tackling the splitting function for the high dimensional feature data and its independent training and evaluation of the trees. Therefore, random forests can be easily implemented with parallel technology on multiple CPU cores. Secondly, random forest works with a divide-and-conquer strategy, which enables it to tackle multi-class problems efficiently. This divide-and-conquer scheme separates the feature data by all the split nodes during training and learns prediction models locally in the leaf nodes. This scheme can effectively handle feature data with multinomial probability distributions. Moreover, random forest holds its good generalization property stemmed from its inherent random sampling and the multiple expert decision scheme, which can avoid the "over fitting" problem bothered in many machine algorithms. Finally, deep learning has been emerging as a hot research topic and has successfully been applied to many problems, random forest can be combined with deep networks, as cascaded frameworks, to further exploit the research areas.

1.3 Statements of Originality

The following contributions reported in this thesis are claimed to be original.

a. A simple, effective and discriminative random forest generated feature is proposed. This intimacy definition feature (IDF) is computed by two members' degree of intimacy (DoI) in a full binary tree, which can encode the path from the root to a leaf node as a floating-point number. With this proposed light feature, we explored the random-forest based cascaded regression model and designed an efficient and accurate face aligner.

b. A novel framework, which takes the advantages of both regression-based methods and constrained local model (CLM), has been proposed. In this joint framework, the conventional local-response maps, which are computed by using random-forest-based regressors, can be computed efficiently and cover a large area for predicting the landmark locations. On the other hand, this model can also be considered as a random-forest regression-based framework with a PDM shape constraint. We named this new model as an efficient likelihood Bayesian constrained local model (elBCLM).
c. A novel random-forest scheme, namely Joint Maximum Purity Forest (JMPF), has been proposed, which can work for classification, clustering, and regression tasks with remarkable performance when compared to conventional random-forest models. In the JMPF scheme, the original feature space is transformed into a compactly pre-clustered feature space, via a trained rotation matrix. The rotation matrix is obtained through an iterative quantization process, where the input data belonging to different classes are clustered to the respective vertices of the new feature space with maximum purity. We have also applied JMPF to image super-resolution, because the transformed, compact features are more discriminative to the clustering-regression scheme.
d. A novel feature-augmented random forest (FARF) model has been presented for image super-resolution, in which the conventional gradient-based features are augmented with gradient magnitudes and different feature recipes are formulated at different stages in a random forest. There are new features in our method. Firstly, gradient magnitudes are employed to enhance the feature discriminative property. Secondly, generalized locality-sensitive hashing (LSH) is used to replace principal

component analysis (PCA) for feature dimensionality reduction. Finally, we presented a generalized weighted ridge regression (GWRR) model for the leaf-node' regressors.

1.4 Outline of the Thesis

This thesis is organized into seven chapters, which are outlined as follows.

Chapter 2 gives an overview of machine learning, random forest, face alignment and image super-resolution techniques and their developments.

Chapter 3 presents a cascaded face alignment model via intimacy definition feature (IDF) to localize the important facial feature points on human faces, which can provide crucial information about face structure and help with face alignment needed for applications like face recognition and face animation, etc.

Chapter 4 introduces a novel constrained local model (CLM) framework, which is called efficient likelihood Bayesian constrained local model (elBCLM). This is the first model which connects two schools of face alignment into a single framework.

Chapter 5 presents a joint maximum purity random forest model for classification, clustering and regression, and this is also applied to the image super-resolution task.

Chapter 6 introduces a feature-augmented random forest model for the image super-resolution task, where some are extension works of chapter 4. In the new model, also some new features, such as principal component analysis (PCA) is replaced by locality-sensitive hashing (LSH) and a generalized weighted ridge regression (GWRR) model is presented for regressors in the leaf-nodes.

Finally, we give conclusions of our proposed work in Chapter 7, where some suggestions for further work are also provided.

Chapter 2. Literature Review

In this chapter, firstly we will review the various machine-learning algorithms, and their main categories. Then, the general concepts, techniques for random forest, face alignment, and image super-resolution are introduced. Finally, the main categories and development of face alignment and image super-resolution are reviewed.

2.1 Machine Learning

Machine Learning is a part of artificial intelligence (AI), which attempts to build up an intelligent system by training its parameters from input data. For example, a machine-learning-based system has been trained to check whether a given E-Mail is a spam or not. In the training process, the system is fed with labelled spam E-Mails and non-spam E-Mails. After training, the system can be used to distinguish spam and non-spam E-Mails for new E-Mails. As illustrated in this example, generalization is the core property of machine learning, which enables the machine-learning-based systems to perform well on unseen data instances.

2.1.1 Supervised Learning

Supervised learning is the machine learning task of inferring a function from labelled training data. A vivid example of supervised learning can be seen in Fig. 2-1, that all the pupils are learning lessons in a classroom which are supervised by a teacher. In supervised learning, the system is fed with labelled training data, i.e., each instance of the training data is consisted of a vector of features and a labelled expected output value. A supervised learning algorithm is designed to produce an inferred function, i.e., a classifier. The trained classifier can predict the labels for the unseen data since it has the powerful capability of generalization. Following are the most widely used supervised learning algorithms:

- ❖ Support Vector Machine (SVM) [71]
- ❖ Linear Regression [152]
- ❖ Logistic regression [117]
- ❖ Naive Bayes [153]
- ❖ Linear discriminant analysis (LDA) [154]
- ❖ Decision trees [155]
- ❖ K-nearest neighbor (K-NN) [156]
- ❖ Neural Networks (Multilayer perceptron) [125]

Fig. 2-1: Pupils are learning lessons in a classroom supervised by their teacher. This image comes from internet of US Department of Education: https://www.flickr.com/photos/departmentofed/9599312337)

2.1.2 Unsupervised Learning

Unsupervised learning is the machine learning task to train an inferred function which can describe the structure from unlabelled given data, i.e., the data has not been classified or categorized. Since the given data has no labels (no error or reward signal), the accuracy of the structure cannot be evaluated in a straightforward way, which distinguishes unsupervised learning from supervised learning and reinforcement learning. In statistics domain, unsupervised learning is mainly employed to solve the problem of density estimation [127]. Moreover, unsupervised learning contains techniques that try to explore and analysis the key features of the data, which are widely used on data mining and data pre-process. The widely used unsupervised learning algorithms including:

- ❖ Clustering
  - ➢ K-means, [129]
  - ➢ Gaussian Mixture Models (GMM) [151]
  - ➢ Hierarchical clustering [128]
- ❖ Neural Networks
  - ➢ Autoencoders [144]
  - ➢ Deep Belief Nets [145]
  - ➢ Generative Adversarial Networks [43]
- ❖ Approaches for learning latent variable models
  - ➢ Expectation–Maximization algorithm (EM) [146]
  - ➢ Method of moments [130, 131]
  - ➢ Blind signal separation techniques
    - ▪ Principal Component Analysis (PCA) [150]
    - ▪ Independent Component Analysis (ICA) [149]
    - ▪ Non-negative Matrix Factorization (NMF) [147]
    - ▪ Singular Value Decomposition (SVD) [148]

2.1.3 Semi-Supervised Learning

Semi-supervised learning problems are on the conditions that among a large amount of input data, only some of the data are labelled. These semi-supervised learning problems are sit in between supervised and unsupervised learning as discussed in above sections. A good example to these problems is a photo archive, in which only a small quantity of the images are labelled, and the majority are unlabelled. Since labelling sample data is time-consuming, expensive, or required for domain experts, on the other hand, unlabelled data is easy and cheap to collect and store, semi-supervised learning algorithms are mainly employed on many real-world machine learning problems. In semi-supervised learning problems, unsupervised learning techniques can be used to explore the structure of the input data. And supervised learning techniques can be employed to make predictions on the unlabelled data, then the data is fed back into the supervised learning algorithm as training data and the fine-tuned model is used to make predictions on new unseen data.

2.1.4 Reinforcement and Deep Learning

2.1.4.1 Reinforcement Learning

Reinforcement learning is an area of machine learning, inspired by behaviorist psychology, concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward. Machine learning (ML) is based on probability theory and statistics [133] and optimization, is the basis for big data, data science [134], predictive modeling, data mining, information retrieval, and so on. Reinforcement learning (RL) is a methodology on machine learning which is widely used on kin to optimal control [135], and operations research. RL to train an *agent* to perform a *task* within an environment as described in Fig. 2-2. The *agent* to perform a *task* within an *environment*. To accomplish a task, the agent will sequentially take an *action* based on the *state* of the environment and the current *policy*; Then, the agent will learn and update its *policy* based on the feedback from the environment in the form of a *reward*; Finally, the agent is able to learn an optimal *policy* after episodes of learning with a *trial-and-error* strategy [132].

As shown in Fig. 2-2, for each time step t, the agent observes a state $s_t \in \mathcal{S}$ which represents the environment, then takes an action $a_t \in \mathcal{A}$ based a policy $\pi: \mathcal{S} \rightarrow \mathcal{A}$. The agent later will receive a reward r_{t+1} for taking action a_t then arrive to the next state: s_{t+1}. Normally the environment is assumed as a Markov Decision Process (MDP), i.e. the probability of the arriving next state s_{t+1} and receiving reward r_{t+1} only depend on current state s_t and action a_t, are unconcerned about any of the previous states or actions.

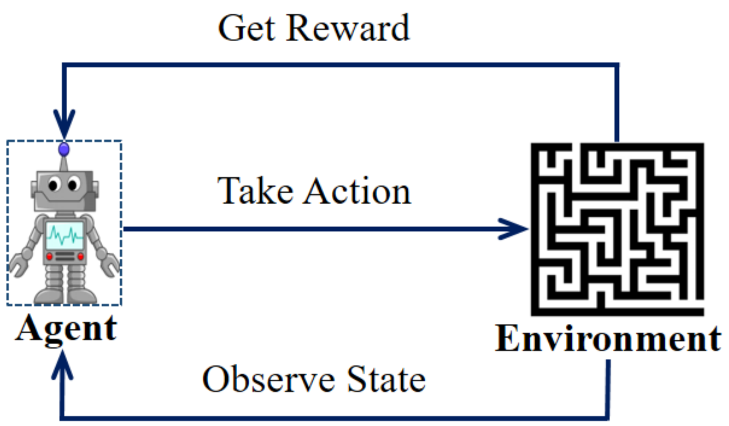


Fig. 2-2: In the reinforcement learning (RL) scheme, the agent takes the observed state as input, then takes an action based on current state and gets a reward from the environment as a feedback. The goal of the agent is to maximize the total reward after a series of actions.

As a long-term perspective policy must take consider of not only the immediate reward but also the final target, so the RL tries to achieve an optimal final expected return via the action-value function $Q^{\pi}(s,a)\colon(\mathcal{S},\mathcal{A})\rightarrow\mathbb{R}$, which is the discounted expected return of rewards given the state, action, and policy,

$$Q^{\pi}(s,a)=\mathbb{E}_{\pi}[R_{t+1}|s_t=s,\ a_t=a\,] \tag{2.1}$$

where

$$R_{t+1}=r_{t+1}+\sum_{k=1}^{T-t-1}\gamma^k r_{t+1+k}\,,0<\gamma\leq 1, \tag{2.2}$$

T is the maximum number of episodes and the expectation is $\mathbb{E}_{\pi}$ is achieved by taking policy π. The optimal action-value function, $Q^{*}(s,a)=max_{\pi}Q^{\pi}(s,a)$ satisfies the Bellman equation [138],

$$Q^{*}(s,a)=\mathbb{E}_{\pi^{*}}\left[r_{t+1}+\gamma\max_{a'\in\mathcal{A}}Q^{*}(s_{t+1},a')\,\middle|\,s_t=s,\ a_t=a\right] \tag{2.3}$$

which provides a natural update rule for learning.

2.1.4.2 Deep Learning

Deep learning (also known as deep structured learning) is a family of machine learning methods that contrasts with "shallow" learning, such as traditional neural network and other machine learning algorithms, e.g., support vector machine (SVM) [71] which contains only input layer and output layer and the input may be transformed with manual feature engineering before training. In deep learning, between input and output layers, one or more hidden layers can be configured. A feedforward deep neural network or multilayer perceptron (MLP) is to map a set of input values to output values with a mathematical function formed by composing many simpler functions at each layer. A convolutional neural network (CNN) [140] is a feedforward deep neural network, with convolutional layers, pooling layers and fully connected layers [142]. A recurrent neural network (RNN) [137] is often used to process sequential input data, with hidden units to store history of past elements. As it is hard for RNN to store information for very long time and the gradient may vanish after long steps, long-short-term-memory networks (LSTM) [136] and gated recurrent unit (GRU) were proposed to address such issues, with gating mechanisms to manipulate information through recurrent cells.

The key advantage of deep learning is its power showing on combating the exponential challenges of the curse of dimensionality. The notion of end-to-end training refers to that a learning model uses raw inputs without manual feature engineering to generate outputs, e.g., CNN based Alex-Net [140] with raw pixels for image classification, Seq2Seq [141] with raw sentences for machine translation, and deep Q-network (DQN) [139] with raw pixels and score to play games.

2.1.4.3 Deep Reinforcement Learning

Recently, deep learning has seen exciting successes in solving reinforcement learning (RL) problems, such as the recent use of a deep Q-network (DQN) in Q-learning to solve Atari games [139], text games [143], and in particularly, the event that Google DeepMind's AlphaGo algorithm [116] beats world class Go players in the year 2016. For an RL system, there is less domain knowledge can be used, so deep learning is used to extract and track the hidden states from partially observable environments.

In these works [139, 143], an RNN [137] (i.e., an LSTM [136] module) is used to represent a Q-function, $Q(s,a;\theta)$, with parameters θ. These models are called as RL-RNN and RL-LSTM, respectively. The parameters θ of Q-learning can be updated based on the observed transitions (s,a,r,s') by follows:

$$\theta \leftarrow \theta + \eta\big(r + \gamma \max Q(s',a') - Q(s,a)\big)\nabla_\theta Q(s,a) \tag{2.4}$$

where s' is the new state and a' is its corresponding action. η , r and γ are hyperparameters, and ∇_θ are the gradients (or Jacobi matrix) of parameters θ.

2.1.3 Ensemble Learning

As there is no machine learning based model that can consistently outperform other models in all problems, an intuitive idea is to combine lots of diverse models into a so-called ensemble learning scheme to reduce the variability of models. A typic example like ensemble learning in real-world is the scene that a group of judges (experts) give scores independently in a sport game as shown in Fig. 2-3, and finally the gymnast's performance will be evaluated with the average score by all the judges.

There are three classic ensembles algorithms in machine learning: bagging, boosting and stacking [118]. The bagging algorithm tries to create numerous diverse models by dividing the training dataset, either randomly selecting a subset from training samples or randomly selecting a subset of features as working candidate features, and make a final decision using majority voting by combining all the models. Random forest [14] is an example of bagging ensemble algorithms. Different to bagging where all the models are created independently, boosting uses another way to combine models. Boosting tries to add models into an existing set of models based on the new model can get better performance than the existing models on the training samples. Gradient boosting machines [119] and AdaBoost [126] are two classic examples of boosting algorithms. The variance and bias of a model [120] can be reduced through boosting algorithm, but it is susceptible to noise data [121].

Fig. 2-3: A group of judges are watching a gymnast in a sport game. Each judge will give a score independently based on the gymnast's performance, and the gymnast normally will get a final average value based all the judges' given scores.

2.1.3.1 Gradient Boosting Machines

Gradient boosting machines algorithm [119] is a boosting algorithm that desires for minimizing a loss function, which measures the difference between the observed and predicted values, by combining a sequence of base learner (weak learner) models (e.g., decision trees). Gradient decent is the common optimization method, which searches the minimum point by going down a gradient at each step. Gradient boosting machines sequentially add a new base learner into the ensemble sequence, so that the new model can get the negative of the loss function's gradient calculated using the current ensemble sequence predictions. Gradient boosting machines have been successfully applied in a range of classification tasks but are also known to be sensitive to noise data.

2.1.3.2 AdaBoost

AdaBoost is the most popular boosting algorithm, which is a supervised learning algorithm for binary classification task. Many works by Freund and Schapire can be considered to contribute to this popular algorithm [122, 124, 120]. A very recent and broad summary of boosting, including AdaBoost, can be found in [123]. Some of the most popular and recent works include the work of Viola & Jones [126] on AdaBoost for face detection. The Viola & Jones detector is one of the most elegant algorithms for face detection that has been quite popular over the recent years. This algorithm can achieve impressive detection rates at high speed.

2.1.3.3 Random Forest

Random forest is another ensemble learning technique which can be employed for working on either classification or regression. Conventional random forest algorithm [14] try to get an average prediction through multiple diverse decision tree. Formally, the training process of random forest involves learning a lot of diverse decision trees, and each tree works as an independent expert. Then the final prediction is come from the majority voting of all the decision trees. In the following section, random forest will be detailed.

2.2 Random Forest

A random forest is an ensemble of binary decision trees $\mathcal{T}^t(x): V \rightarrow R^d$, where $t = 1, \cdots, T$ is the index of the trees, x is a sample from the m-d feature space: $V \in R^m$, and $R^d = [0,1]^d$ represents the space of class probability distributions over the label space $Y = \{1, \ldots, d\}$.

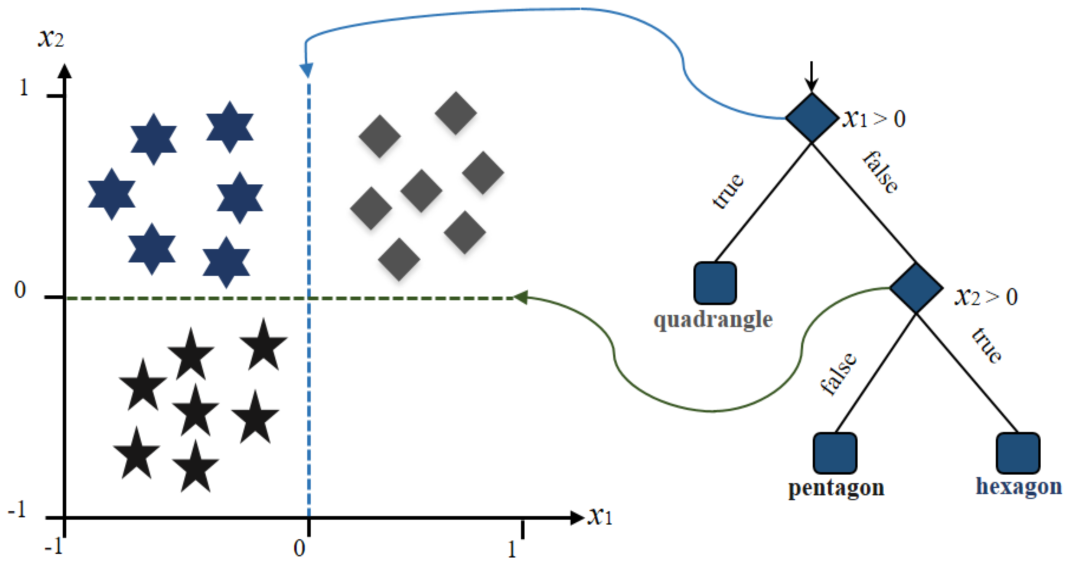


Fig. 2-4: Random forest for clustering data.

As shown in Fig. 2-4, the vertical dotted line forms a hyperplane: x_1=0, chosen at the first split node, i.e. the root node, to separate all the training samples, and the horizontal dotted line is the hyperplane: x_2=0, for the second split node to cluster all the feature data assigned to this node. This results in separating the three data samples (quadrangle, pentagon and hexagon) into three leaf nodes.

In the inference stage, each decision tree returns a class probability $p_t(y|\boldsymbol{v})$ for a given test sample $\boldsymbol{v} \in R^m$, and the final class label y^* is then obtained via averaging, as follows:

$$y^* = \arg\max_{y} \frac{1}{T} \sum_{t=1}^{T} p_t(y|\boldsymbol{v}). \tag{2.5}$$

A splitting function $s(\boldsymbol{v};\Theta)$ is typically parameterized by two values: (i) a feature dimensional index: $\Theta^i \in \{1, \ldots, m\}$, and (ii) a threshold $\Theta^t \in \mathbb{R}$. The splitting function is defined as follows:

$$s(\boldsymbol{v};\Theta) = \begin{cases} 0, & \text{if } \boldsymbol{v}(\Theta^i) < \Theta^t, \\ 1, & \text{otherwise,} \end{cases} \tag{2.6}$$

where the outcome defines to which child node $\boldsymbol{v}$ is routed, and 0 and 1 are the two labels belonging to the left and right child node, respectively. Each node chooses the best splitting function Θ^* out of a randomly sampled set $\{\Theta^i\}$, and the threshold Θ^t is determined by optimizing the following function:

$$I = \frac{|L|}{|L|+|R|} H(L) + \frac{|R|}{|L|+|R|} H(R), \tag{2.7}$$

where L and R are the sets of samples that are routed to the left and right child nodes, respectively, and $|S|$ represents the number of samples in the set S. During the training of an RF, the decision trees are provided with a random subset of the training data (i.e. bagging), and are trained independently. Training a single decision tree involves recursively splitting each node, such that the training data in the newly created child node is clustered conforming to class labels. Each tree is grown until a stopping criterion is reached (e.g. the number of samples in a node is less than a threshold or the tree depth reaches a maximum value) and the class-probability distributions are estimated in the leaf nodes. After fulfilling one of the stopping criteria, the density model $p(y)$ in each leaf node is estimated by using all the samples falling into the leaf node, which will be used as a prediction of class probabilities in the inference stage. A simple way to estimate the probability distribution function $p(y)$ is by averaging all the samples in the leaf node, and there are many variants, such as fitting a Gaussian distribution, kernel density estimation, etc.

In (9), $H(S)$ is the local score for a set of samples in S (S is either L or R), which is usually calculated by entropy, as shown in Eqn. (2.8), and it can be replaced by variance [18, 51, 1] or by the Gini index [14].

$$H(S) = -\sum_{k=1}^{K} \left[p(k|S) * \log\big(p(k|S)\big) \right], \tag{2.8}$$

where K is the number of classes, and $p(k|S)$ is the probability for class k, which is estimated from the set S. For the regression problem, the differential entropy is used, and is defined as,

$$H(q) = \int_y \; q(y|x) * \log(q(y|x))d_y, \tag{2.9}$$

where $q(y|x)$ denotes the conditional probability of a target variable given an input sample. Assuming that $q(.,.)$ is of Gaussian distribution, and has only a set of finite samples, the differential entropy can be written as,

$$H_{Gauss}(S) = \frac{K}{2}(1 - \log(2\pi)) + \frac{1}{2}\log(\det(\Sigma_S)), \tag{2.10}$$

where $\det(\Sigma_S)$ is the determinant of the estimated covariance matrix of the target variables in S.

RF-based approaches hold some properties, which make them powerful classifiers as SVM (support vector machine) [70] and AdaBoost (short for "Adaptive Boosting") [73]. Both SVM and AdaBoost work as to approximate the Bayes decision rule – known to be the optimal classifiers – via minimizing a margin-based global loss function.

2.3 Face Alignment

Face alignment is a classic computer vision task which aligns the facial components, e.g., eye, nose, mouth, and contour in a given face image. Face alignment is very important for lots of applications, such as face recognition, facial animation, emotion recognition, face hallucination, 3D face reconstruction, and so on. Face alignment assumes that a face bounding box is given, this can be done by any face-detection algorithm, such as the Viola-Jones [12] face detector, or by manual annotations. Facial landmarks that represent face shape can then be estimated by alignment methods. Traditional methods, such as the active shape model (ASM) [3] and active appearance model (AAM) [4] are statistical models. ASM represents the shape of an object, while AAM represents both texture and shape. Constrained local models (CLM) [24, 25, 27, 28, 31] attempt to model shape prior to integrate with local texture. It assumes face local appearance and global face-shape patterns lie in a linear subspace spanned by the bases learned from principal component analysis (PCA). In [26], a face-shape fitting process is formulated as a non-linear optimization problem by minimizing the misalignment error (i.e. the average distance of all the respective landmarks normalized by the inter-pupil distance) between the model instance and a given image. The model parameters that control the shape and appearance variations of faces are hence learned from that optimization. In [26], an extension to the inverse compositional image-alignment algorithm [29] was proposed, which decouples shape information from appearance. This method [29] forms a computationally efficient AAM framework. A CLM model is usually composed of three main parts: a point distribution model (PDM), patch experts which perform matching for local patches around landmarks of interest, and a final fitting process. Different fitting strategies have been used in CLM variants. Regularized landmark mean shift (RLMS) [28] is a popular strategy, which estimates the rigid and non-rigid parameters by minimizing the misalignment error of landmarks, regularized by overly complex or unlikely shapes. In [27], a local neural filed (LNF) patch expert was proposed, which learns the similarity and long-distance sparsity constraints to derive relationships between neighboring pixels and longer-distance pixels. This method [27] achieves state-of-the-art performance, compared to the traditional CLM based methods. In [32], the authors proposed an exemplar-based graph matching (EGM) framework for face alignment, in which the response mappings of all the facial landmarks are fitted by selecting from a pool of training exemplar poses.

However, these models have limited expressive power to capture all possible complex and subtle face features, due to variations in expression, illumination, pose, etc. Furthermore, due to the intensively computing from the inverse of Hessian matrix and the Jacobian matrix [6, 27, 28, 29, 32], it is very hard to improve the speed of those CLM-like algorithms exponentially.

Recently, deep learning based models have been emerging as hot research topics and successfully applied to many computer vision tasks such as generic object detection and classification [33, 34, 35], handwritten digit recognition [38], RGB-D object recognition [39], image super-resolution [41, 42, 43], visual tracking [44], face alignment [36, 37, 40, 46] and so on. In [40], the authors try to improve face landmark detection through multi-task learning by designing a tasks-constrained deep model with task-wise early stopping criterion to increase the learning convergence rate. Authors in [37] exploit using deep neural network to learn feature-to-pose mapping functions by combining the cascaded framework for regressing pose-indexed features. To solve inefficiency issue appears in above mentioned works, paper [46] proposed an eight-learnable-layer deep convolutional neural network (DCNN) with rectified linear unit (ReLU) rather than tanh activation function, which achieves five times faster in training convergence without decreasing its accuracy. To better initialize facial poses, in [36], the authors present a global exemplar-based deep auto-encoder network (GEDAN) to increase the capability of handling large pose variations by incorporating several exemplars at the top layer in a non-linear fashion. Although these brute-force-style deep learning approaches has achieved promising performance in terms of fitting accuracy, but their heavy computation is a big obstacle to real-world applications when there is less chance with graphics processing unit (GPU) for speed-up or somewhere the hardware resources are limited, such as mobile devices.

Thus, an accurate, real-time and small size model based face alignment algorithm is eager called from real-world industries, such as smart mobile phone applications.

2.4 Image Super-Resolution

Image super-resolution (SR), which recovers a high-resolution (HR) image from one single image or a few low-resolution (LR) images, has been a research hotspot in the field of image processing for decades. SR is a well-known ill-posed problem, which needs elaborate techniques from mathematics and machine learning. Prior methods on SR are mainly based on edge preserving techniques, such as New Edge-directed Interpolation (NEDI) [60], Soft-decision Adaptive Interpolation (SAI) [59], Directional Filtering and Data-Fusion (DFDF) [58], Modified Edge-Directed Interpolation (MEDI) [57], etc.

The neighbor-embedding (NE) methods [89, 90] set the milestone on the patch-learning-based super-resolution approach. In this approach, each LR patch is approximated as a linear regression of its nearest LR neighbors in a collected dataset, while its HR counterpart can be reconstructed with the same coefficients of corresponding HR neighbors, based on the non-linear manifold structure. Although the NE method is simple and practical, it requires a huge dataset (millions of patches) to achieve good reconstruction quality and it is computationally intensive, because the *k*-NN (k-nearest neighbour) algorithm [8, 19] is used in searching neighboring patches in the huge dataset. Instead of using the patches extracted directly from natural images, Yang *et al*. [88] employed sparse coding [72,88] to represent patch images, of large size, efficiently, which opens the era for sparse coding in the image inverse problems.

The sparse-coding super-resolution (ScSR) approach is a framework that the HR counterpart of an LR patch can be reconstructed aided by two learned dictionaries, with the sparse constraint on the coefficients via the following formulations:

$$y \approx \mathrm{D}_l\alpha, x \approx \mathrm{D}_h\alpha, \quad \alpha \in \mathbb{R}^k \text{ with } \|\alpha\|_0 \ll k. \tag{2.11}$$

The compact LR and HR dictionaries can be jointly learned with a sparsity constraint, using the following sparse representation:

$$\mathrm{D}_h, \mathrm{D}_l = \underset{\mathrm{D}_h,\mathrm{D}_l}{\operatorname{argmin}} \|x - \mathrm{D}_h\alpha\|_2^2 + \|y - \mathrm{D}_l\alpha\|_2^2 + \lambda\|\alpha\|_0, \tag{2.12}$$

where y and x are the LR patch and the corresponding HR patch, respectively; and D_l and D_h are the LR and HR dictionaries learned from the LR and the corresponding HR patch samples, respectively. The value of k in $\|\alpha\|_k$ is the sparsity factor of the coefficients α. $\|\alpha\|_0$ is l^0-norm, which means the non-zero count of the coefficients in α. For each LR patch y of an input LR image Y, the problem of finding the sparse coefficients α can be formulated as follows:

$$\min\|\alpha\|_0 \text{ s.t. } \|\mathrm{D}_l\alpha - y\|_2^2 \leq \varepsilon \tag{2.13}$$

or

$$\min\|\alpha\|_0 \text{ s.t. } \|F\mathrm{D}_l\alpha - Fy\|_2^2 \leq \varepsilon, \tag{2.14}$$

where F is a linear or non-linear feature-extraction operator on the LR patches, which makes the LR patches more discriminative from each other. Typically, F can be chosen as a high-pass filter, and a simple high-pass filter can be obtained by subtracting the input from the output of a low-pass filter, as in an early work [75]. In [62, 64, 65, 68], first and second-order gradient operators are employed on up-sampled versions of low-resolution images, then four patches are extracted from these gradient maps at each location, and concatenate them to become feature vectors. The four 1-D filters used to extract the derivatives are:

$$\left.\begin{array}{r} F_1 = [-1, 0, 1],\ F_2 = {F_1}^T \\ F_3 = [1, 0, -2, 0, 1],\ F_4 = {F_3}^T \end{array}\right\} \tag{2.15}$$

The ideal regularization term for the sparse constraint on the coefficients α is the l^0-norm (non-convex), but, based on greedy matching, it leads to an NP-hard problem. Alternatively, Yang et al. [88] relaxed it to l^1-norm, as shown in the following formulation:

$$\min\|\alpha\|_1 \text{ s.t. } \|F\mathrm{D}_l\alpha - F\mathrm{y}\|_2^2 \leq \varepsilon. \tag{2.16}$$

The Lagrange multiplier provides an equivalent formulation as follows:

$$\min_{\alpha}\|F\mathrm{D}_l\alpha - F\mathrm{y}\|_2^2 + \lambda\|\alpha\|_1, \tag{2.17}$$

where the parameter λ balances the sparsity of the solution and the fidelity of the approximation to y. However, the effectiveness of sparsity was challenged in [3, 65], as to whether real sparsity can help image classification and restoration, or locality property can achieve the same effect. Timofte et al. [62] proposed an anchored neighborhood regression (ANR) framework, which relaxes the sparse decomposition optimization (l^1-norm) of [64, 88] to a ridge regression (l^2-norm) problem.

An important step in the ANR model is the relaxation of the l^1-norm in Eqn. (2.17) to the l^2-norm least-squares minimization constraint, as follows:

$$\min_{\alpha}\|F\mathrm{D}_l\alpha - F\mathrm{y}\|_2^2 + \lambda\|\alpha\|_2, \tag{2.18}$$

where D_l and D_h are the LR and HR patch-based dictionaries, respectively. This l^2-norm constraint problem can be solved with a closed-form solution from the ridge regression

[76] theory. Based on the Tikhonov regularization/ridge-regression theory, the closed-form solution of the coefficients is given:

$$\alpha = (D_l^T D_l + \lambda I)^{-1} D_l^T F y. \quad (2.19)$$

We assume that the HR patches share the same coefficient α from their counterpart LR patches, i.e., $x = \mathrm{D}_h \alpha$. From Eqn. (2.19), we have:

$$x = \mathrm{D}_h (D_l^T D_l + \lambda I)^{-1} D_l^T F y. \quad (2.20)$$

Therefore, the HR patches can be reconstructed by: $x = P_G F\mathrm{y}$, where P_G can be considered a projection matrix, which can be calculated offline, as follows:

$$P_G = \mathrm{D}_h (D_l^T D_l + \lambda I)^{-1} D_l^T. \quad (2.21)$$

Ridge regression allows the coefficients α to be calculated by multiplying the constant projection matrix P_G with the new extracted feature Fy, as described in Eqn. (2.20) and Eqn. (2.21). More importantly, the projection matrix P_G can be pre-computed, and this offline learning enables significant speed-up at the prediction stage.

Timofte *et al*. [65] further extended the ANR approach to the A+ approach, which learns regressors from all the training samples, rather than from a small quantity of neighbors of the anchor atoms as ANR does. Later, there are numerous variants and extended approaches, based on ANR and A+ [3, 78, 82, 93, 95, 81, 69, 74]. By investigating the ANR model, Li et al. [3] found that the weights of the supporting atoms can be of different values to represent their similarities to the anchor atom. Based on this idea, the normal collaborative representation (CR) model in ANR is generalized to a weighted model, named as weighted collaborative representation (WCR) model, as follows:

$$\min_{\alpha} \|F\mathrm{D}_l \alpha - F\mathrm{y}\|_2^2 + \|\lambda_{WCR} \alpha\|_2, \quad (2.22)$$

where λ_{WCR} is a diagonal matrix. The weights on the diagonal atoms are proportional to their similarities to the anchor atom. Similarly, the new closed-form solution for the coefficients can be calculated offline, as follows:

$$\alpha^* = (D_l^T D_l + \lambda_{WCR})^{-1} D_l^T F\mathrm{y}, \tag{2.23}$$

and the new projection matrix is given as follows:

$$P_G^* = \mathrm{D}_h (D_l^T D_l + \lambda_{WCR})^{-1} D_l^T. \tag{2.24}$$

The WCR model can further improve the ANR or A+ model in terms of image quality, but it is still a time-consuming problem to find the most similar anchor atoms in a dictionary, and this always hinders its applications where fast speed is greatly required.

Schulter *et al.* [18] adopted the random forest as a classifier, and the regressors are learned from the patches in the leaf-nodes. With the same number of regressors, these random-forest-based methods [8, 41, 42, 43] can perform on a par with the A+ method in terms of accuracy. However, they achieve an increase in speed, because the sublinear search property of random forest can remarkably reduce the regressors' search complexity.

Recently, deep learning has become another research hotspot, which has been successfully applied to image super-resolution [18, 19, 77, 79] and achieved promising performance, particularly in terms of image quality. In [18, 19], a convolutional neural-network-based image super-resolution (SRCNN) was proposed, in which an end-to-end mapping between LR and HR images is learned through a deep convolutional neural network (CNN). [42] presented a super-resolution approach with very deep networks with extremely high learning rates, and the deep network convergence rate is sped up by residual learning. Meanwhile, [43] presented a generative adversarial network (GAN)-based deep residual network model for image super-resolution (SRGAN), in which content loss and adversarial loss are combined as an image perceptual loss function. The proposed deep residual network in [43] can super-resolve photo-realistic textures from 4-times down-sampled images, and an extensive mean-opinion-score (MOS) criterion is proposed to test the perceptual quality gained by using the SRGAN approach. Although deep-learning-based approaches can achieve superior performance compared to other SR methods, their heavy computation is always a big obstacle to their extensive applications with real-time requirements, where the graphics processing unit (GPU) may not be available, such as smart mobile phones.

2.5 Conclusions

This chapter serves as a review of machine learning, a survey of the principles and development of random forest, face alignment and image super-resolution techniques. Some well-known face alignment techniques and image super-resolution have also been reviewed, and the drawbacks of them are also discussed. In the following chapters, some proposed methods based on random forest on these two research topics will be presented.

Chapter 3. Cascaded Face Alignment via Intimacy Definition Feature

Recent years have witnessed the emerging popularity of regression-based face aligners, which directly learn mappings between facial appearance and shape increment manifolds. In this chapter, we propose a random-forest based, cascaded regression model for face alignment by using a novel locally lightweight feature, namely intimacy definition feature (IDF). This feature is more discriminative than pose-indexed feature, more efficient than histogram of oriented gradients (HOG) feature and scale-invariant feature transform (SIFT) feature, and more compact than the local binary feature (LBF). Experimental validation of our algorithm shows our approach obtains state-of-the-art performance when testing on some challenging datasets. Compared with the LBF based algorithm, our method achieves about two times in speed, 20% improvement in terms of alignment accuracy and save an order of magnitude on memory requirement.

3.1 Introduction

Face alignment is a process to locate key-points and facial feature (like eyebrows, eye corners, and mouth corners, see Fig. 3-1) from a given face image. Face alignment is an active research topic in computer vision. It is often used as an early, but crucial, step to other important tasks for face analysis, such as emotion and expression recognition [9, 47], face recognition [10], and face hallucination [11, 49, 50]. It is also used in many other applications, such as human-computer interaction (HCI), video conferencing, gaming and animation, etc., and has received intensive attention from the computer-vision research community.

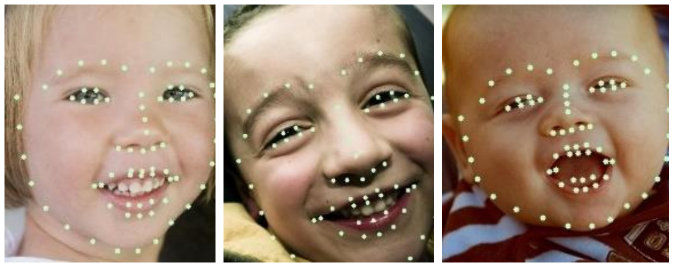


Fig. 3-1 Face alignment fitting results by proposed IDF method with 68 facial points (face images from the Helen dataset [21]).

In the past few years, a new family of face-alignment algorithms, which directly learns regressors from facial appearance to the shape increments, has been emerging [1, 5, 6, 8, 13]. These regression-based methods are gaining popularity, due to their excellent performance and high efficiency in the face-alignment task. Pose-indexed feature [1, 8, 13, 48], which index provides some clue about the hierarchical structure of the shape, is an explored paradigm to boost fitting efficiency due to its simple pixel-intensity comparison. In [6], the handcrafted scale-invariant feature transform (SIFT) feature is used for accurate fitting. Inspired by pioneers' works [1, 5, 6, 8, 13, 48], in this chapter, we propose a novel, discriminative and efficient feature, which can be incorporated into regression based face-alignment frameworks to further boost their performance.

3.2 Random Forests for Face Landmark Alignment

The landmark localization algorithm is important for face recognition and other related applications which requires extraction of local descriptors at some specified feature points or landmarks in a face. For face alignment, numbers of points, e.g., 17 or 68, are selected and searched from a face image. An example of the landmarks is shown in Fig. 3-1, in which 68 facial points are located around the eyes, nose, lips, and face contour. These feature points, which carry the most significant information about a face, are useful for discriminative and generative analysis. Based on these feature points, a model can then be learned from numbers of landmark-labeled face images, used for facial-shape estimation for unseen face images.

Recently, roughly three categories for face alignment are under researching. They are variants of active shape model (ASM) [3] and active appearance model (AAM) [4] with parametric models of appearance, deep learning based models [36, 37, 40, 46] and regression-based models which directly model a mapping from facial pixel appearance to shape increment [1, 5, 6, 8, 13].

The regression-based face alignment approaches tackle face fitting problem by estimating mapping functions between the appearance and the shape increment manifolds. Random forests are employed on regression-based algorithm in order to reduce the regressors' search complexity. In our algorithm, we adopt cascaded shape-regression paradigm that firstly proposed by Dollar et al. [8] and extend the work of LBF [1]. Different from other methods, this approach progressively refines the initial shape in several stages directly from appearance, without learning any parametric shape or appearance models. To illustrate our proposed methods clearly, we firstly give a brief review of the main principles of random forest and cascaded-shape regression in this section.

3.2.1 Random Forest for Clustering

Random forests [14] (RFs) have emerged recently as very useful in many computer vision tasks, including object detection [16], data clustering [17], image super-resolution [18, 19], etc. This method is relatively simple, and has many merits which include: (i) efficiency in both training and prediction stages, (ii) inherent unsupervised classification capability for multi-class problems, (iii) suitability for parallel processing for all the trees, and (iv) good performance on high-dimensional data for classification, regression and clustering tasks.

A random forest is an ensemble of T binary decision trees $T^t(x)\colon V \rightarrow \mathbb{R}^{\mathrm{K}}$, where t is the index of the trees, $V \in \mathbb{R}^{\mathrm{M}}$ is the M-dimensional feature space, and $\mathbb{R}^{\mathrm{K}} = [0,1]^{\mathrm{K}}$ represents the space of class probability distributions over the label space $Y = \{1, \ldots, \mathrm{K}\}$, as shown in Fig. 3-2.

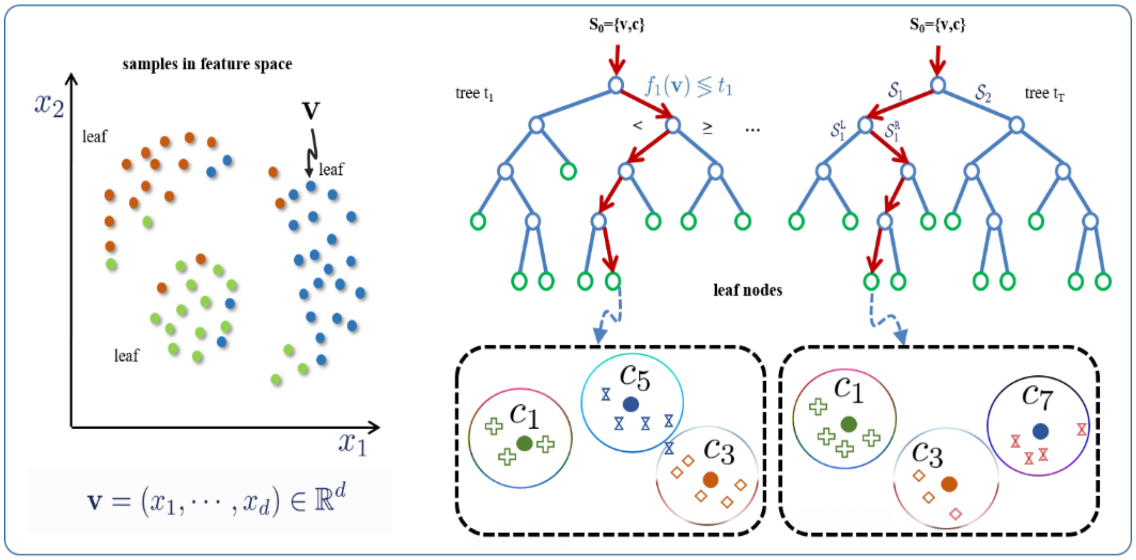


Fig. 3-2 An overview of random-forest based clustering.

In the inference stage, each decision tree returns a class probability $p_t(y|\boldsymbol{v})$ for a given enquiry sample $\boldsymbol{v} \in \mathbb{R}^{\mathrm{M}}$, and the final class label y^* is then obtained via averaging:

$$y^* = \arg\max_{y} \frac{1}{T} \sum_{t=1}^{T} p_t(y|\boldsymbol{v}). \tag{3.1}$$

A splitting function $s(\boldsymbol{v};\Theta)$ is typically parameterized by two values: (i) a feature dimension $\Theta_1 \in \{1, \ldots, \mathrm{M}\}$, and (ii) a threshold $\Theta_2 \in \mathbb{R}$. The splitting function is defined as follows:

$$s(\boldsymbol{v};\Theta) = \begin{cases} 0, & \text{if } \boldsymbol{v}(\Theta_1) < \Theta_2, \\ 1, & \text{otherwise,} \end{cases} \tag{3.2}$$

where the outcome defines to which child node the sample $\boldsymbol{v}$ is routed, and 0 and 1 are the two labels belonging to the left and right child nodes respectively. Each node chooses the best splitting function Θ^* out of a randomly sampled set $\{\Theta^i\}$ by optimizing the following function:

$$I = \frac{|L|}{|L|+|R|} H(L) + \frac{|L|}{|L|+|R|} H(R), \tag{3.3}$$

where L and R are the sets of samples that are routed to the left and the right child nodes respectively, and $|S|$ represents the number of samples in the set S. During the training of a random forest (RF), each decision tree is provided with a random subset of the training data (i.e. bagging), and is trained independently of other trees. Training a decision tree involves recursively splitting each node, such that the training data in the newly created child nodes are clustered conforming to class labels. Each tree is grown until a stopping criterion is reached (e.g. the number of samples in a node is less than a threshold or the tree depth reaches a maximum value), and the class probability distributions are estimated in the leaf-nodes. $H(S)$ is the local score for a set of samples (S is either L or R), which normally is calculated using entropy as in Eqn. (3.4), but it can be replaced by variance [1] or the Gini index [14].

$$H(S) = -\sum_{k=1}^{K} [p(k|S) * \log(p(k|S))] \tag{3.4}$$

where K is the number of classes, and $p(k|S)$ is the probability for class k, which is estimated from the clustered set S.

3.2.2 Cascaded Regression-based Model

Many face alignment methods work under a cascaded framework [1, 5, 6, 8], where an ensemble of N regressors operates in a stage-by-stage manner, which are referred to as stage regressors. This approach was first explored in [8]. At the inference stage, the input to a regressor (R_t) at stage t is a tuple (I, S_{t-1}), where I is an image and S_{t-1} is the shape estimate from the previous stage (the initial shape S_0 is typically the mean shape of the training set). The regressor extracts features with respect to the current shape estimate, and regresses a vector of shape increment as follows:

$$S_t = S_{t-1} + R_t\big(\phi_t(I, S_{t-1})\big), \tag{3.5}$$

where $\phi_t(I, S_{t-1})$ is referred to the feature extraction function, such as the pose-indexed features, i.e. they depend on the current shape estimate. The cascade progressively infers the shape in a coarse-to-fine manner – the early regressors handle large variations in shape, while the later ones ensure small refinements. After each stage, the shape estimate resembles the true shape closer.

In our algorithm, the feature extraction function $\phi_t(I, S_{t-1})$ is the function to generate the local IDF values derived from the pose-indexed feature. There is an observation, proved by intensive experimental results, that the shape increments have close correlation with the local features of the landmarks which define the face shape. Thus, given the features and the target shape increments $\{\Delta S_t = S - S_{t-1}\}$, we can learn the linear projection matrix R_t. Most cascaded regression models [1, 5, 6, 8, 13] share the similar workflow, as shown in Fig. 3-5.

3.3 Intimacy Definition Feature based Cascaded Regression Model

In this section, we will first introduce a novel feature, which is efficient for local pattern representation and matching, based on measuring the degree of intimacy (DoI) value between two members (leaf-nodes) in a binary family tree.

3.3.1 Efficient Metric on Intimacy Definition Feature

To explain the features, we use a family member structure to illustrate the binary tree in random forests scheme, as shown in Fig 3-3. In this structure, each leaf-node represent a family member, and their relationship are measured by DoI value which can be computed by their respective intimacy definition feature (IDF) values. In Fig. 3-3, the DoI value between David and Daniel should be stronger than that between David and Denis. This is because David and Daniel have the same father, while David and Denis do not have the same father but they share the same grandfather only. The way to let the computer learn the DoI value, between any two members in the same generation or level in the hierarchical family tree, is to digitize the DoI values by setting values to nodes and defining a distance metric between any two nodes.

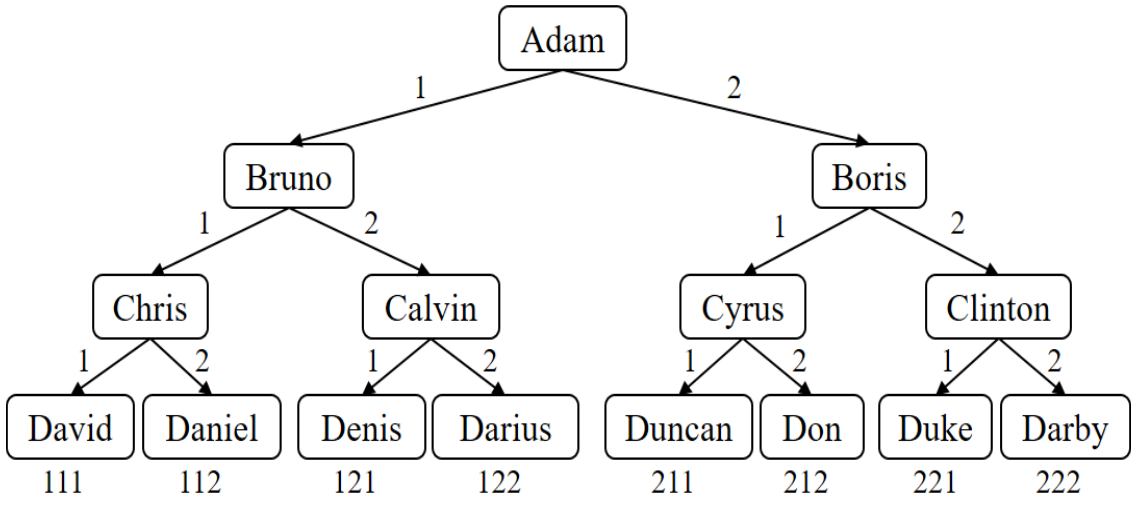


Fig. 3-3 A family tree with the degree of intimacy (DoI) values of family members in 4th generation.

As we can see in the family tree in Fig. 3-3, two persons, who share more recent predecessor, should be more intimate than those who share relatively distant predecessor, as described in the previous example. However, how can a computer know this intimacy, based on this logic comparison operation? In this chapter, we propose a simple, yet efficient, method to compare the DoI values between two family members in the same generation, particularly in the leaf-nodes. We firstly assign two persons in the same generation (same level in the full binary tree) with two small values which indicate they are very close. For example, we set 1 and 2 as the respective *path values* to the two offspring nodes (e.g. David is the younger brother so his *path_value* is 1, while Daniel is the older brother so his *path_value* is 2) in the full binary family tree. Then, we assign a relatively larger value, e.g. 10, to the *generation value k* for each generation level. Each node (except the root node) can then be encoded by summing up all the corresponding level weights along the path from the root to the node of a member concerned, where a level weight of a node is computed by multiplying the value of the node and its corresponding generation value *k*. We name this as the intimacy definition feature (IDF) value of the node (family member), which can be calculated as follows:

$$IDF = \sum_{l=1}^{L} path\_value_l * k^l, \tag{3.6}$$

where L is the total number of levels in the family tree. Therefore, the IDF value of David can be encoded as: 111 ($1\times10^2+1\times10^1+1\times10^0$), and Daniel with IDF value: 112 ($1\times10^2+1\times10^1+2\times10^0$). We can also encode Denis as IDF value: 121 ($1\times10^2+2\times10^1+1\times10^0$). The intimacy distance between David and Daniel is 1 (1 = abs(111−112)), and the distance between David and Denis is 10 (10 = abs(111−121)). The distances show that the intimacy between David and Daniel should be greater than that between David and Denis. Based on the proposed IDF, we can compute the DoI value between the IDF values of any two members in the family tree. Through the family tree as constructed in Fig. 3-3, the family members (nodes) can be replaced by visual features, which are encoded by IDF values. Consequently, the similarity between two family members (nodes) can be measured by computing their DoI value.

In our study, we found that this simple, yet efficient, feature computed by traveling a tree in random forests, which can achieve promising performance, in terms of both accuracy and speed, as shown in Section IV. When using the encoded feature values for linear regression on the leaf-nodes for prediction, for more reliable and better performance, the feature is normalized as follows:

$$nomalized\_IDF = \frac{(IDF - IDF_{min})}{(IDF_{max} - IDF_{min})}, \tag{3.7}$$

where IDF_{min} and IDF_{max} are the minimum and maximum IDF values respectively, in the same level under consideration. Using our example, the range of the IDF values in the binary tree is [100, 222], i.e., IDF_{min} = 100 and IDF_{max}= 222. Therefore, based on Eqn. (3.7), the normalized IDF value for David (**111**) can be calculated as: (111−100)/(222−100) = **0.090164**.

3.3.2 Derive IDF Feature from Pose-indexed Feature in Random Forest

A pose-indexed feature is the value of two pixels' intensity difference. For every landmark point, those two pixels used to compute the pose-indexed value are chosen with two randomness in the random forest splitting rule, which means that they are randomly sampled from the candidate pixel set (e.g. 500) and the threshold is also randomly selected. The positions of the pixel pair and the threshold to be used are decided, based on maximizing the information gain obtained when splitting all the samples in a node into its left and right nodes.

Same as the LBF [1] feature, we discard such learned local pose-indexed features since they are not sufficiently discriminative, or do not encode the path of a sample along a tree explicitly. Instead, we encode the path of a sample along a tree ended at a leaf-node, using our proposed IDF values. As described in Fig. 3-3, each IDF value encoded in a leaf-node is one floating-point scalar, which can achieve highly dimensionality reduction compare to the sparse but high-dimensional binary LBF [1] vector features.

For each stage, the whole feature vector: $\phi_t(I, S_{t-1})$, is a concatenation of a set of independent local-feature, which be used in the mapping functions: $R_t\left(\phi_t(I, S_{t-1})\right)$. All the IDF features are concatenated to form a global feature mapping function Φ_t for learning a global linear projection, i.e. the regressor R_t, in the next step. All the pixel pairs are sampled from the neighborhood area which are centered at landmark points. The idea of pose-indexed feature is described in Fig. 3-4.

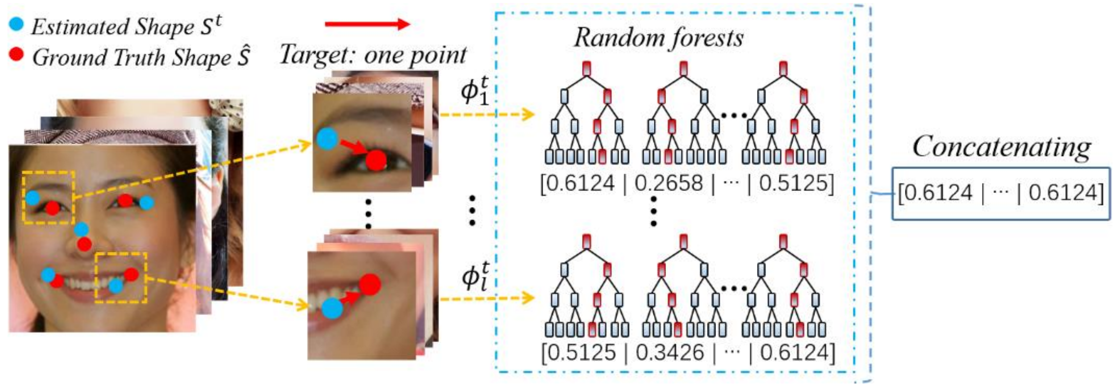


Fig. 3-4 The process of IDF-based feature vector extraction

In both training and inference stages, the neighborhood-size for each landmark can be reduced, when moving from one cascade to another cascade. Therefore, the cascaded shape regression can operate from coarse to fine progressively.

3.3.3 IDF Feature with Regression-based Model

Our proposed algorithm extends from the LBF-based method in [1], which improves the supervised descent method (SDM) [6] used in linear regression. The insight of SDM directly learns shape increment from appearance changes can be viewed to estimate the conditional likelihood function $p(y|x)$, where y and x are shape increment and appearance respectively. From a theoretical perspective, the SDM can be regarded as an extension of the Lucas & Kanade (LK) algorithm [45]. The LK algorithm is worked as a classic optical flow algorithm in early years to tackle image and object alignment problems, which holds an assumption that a linear relationship can be estimated from pixel appearance to geometric displacement.

In [1], random forests were used for training to minimize the alignment error for the respective landmarks with LBF, rather than the pose-indexed feature in the leaf-nodes. LBF is a local feature, which is coded as a sparse binary array, by placing the value '1' for leaf-nodes, where samples fall into them eventually while traversing a tree in random forests, and the value '0' otherwise. Each landmark is coded individually, and the local features are concatenated to form a global feature vector, which is then learned by using ridge regression (i.e., linear regression with L^2 regularization). Rather than using LBF, our proposed IDF is employed in the cascaded alignment framework, as depicted in Fig. 3-5. The success of LBF [1] method is due to its feature-learning step, where features are explicitly learned for the given specific task. Due to the sparse nature of the feature vector of LBF, the inference phase can be reduced to traversing the forest, and performing simple table look-ups and additions. The authors in [1] claimed that LBF method can achieve an impressive speed of approximately 3,000 fps, (with tailored setting on some parameters), in its fast version.

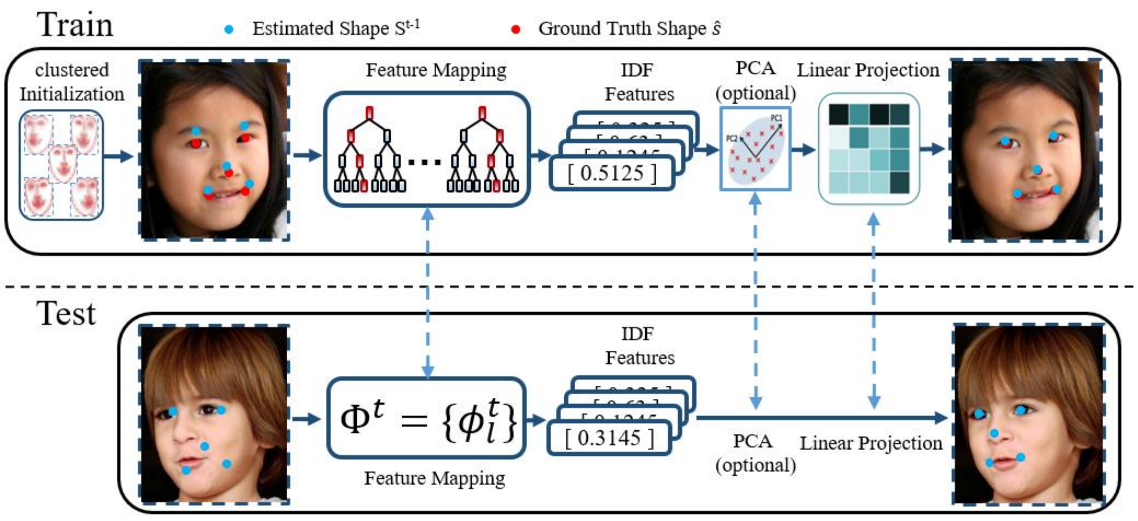


Fig. 3-5 An overview of the workflow for IDF-based cascaded regression face alignment.

However, LBF has a high dimensionality. Assume that the number of landmarks (or forests) for a face is l, the number of trees of a forest is t, and the depth of a tree is d. The dimensionality of LBF will then be $l \cdot t \cdot 2^{(d-1)}$. For a normal setting of $l = 68$, $t = 10$, and $d = 7$, the feature dimension is $68 \times 10 \times 2^{(7-1)} = 43{,}520$, which is relatively high. Usually, with more and deeper trees, the alignment errors will become smaller. However, the high dimensionality of LBF restricts it from using deeper trees. Although the feature is sparse, its high dimensionality imposes a high burden on the computation of linear regression and the storage requirement. An intuitive way to solve the problem is to employ PCA to reduce the dimensionality. But LBF is a binary, sparse feature, and carries labelling information, which makes PCA not acceptable for the feature. To avoid the computational complexity, the LBF-based approach should limit the tree depth to relative small value, e.g.: 5, which means that there are, at most, 16 leaf-nodes in each tree. Consequently, this heavily restricts its capability on classification and regression.

Compared to the pixel pose-indexed feature [13], LBF is more discriminative because it explicitly encodes the full path, from the root to the leaf-node of each sample. Although LBF is discriminative, it is hard to greatly improve its performance because of its high dimensionality when using deeper trees. To improve the performance, an intuitive way is to replace LBF with another more compact and efficient *index feature*, which also encodes the path of a sample along a tree. But the performance is very poor, because index values are similar to labels, which inclines the results to over-fitting. A simple analysis in Fig. 3-3 can help describe the problem of using an *index feature*. Suppose that we simply set the indices for David, Daniel, and Denis at 1, 2, and 3, respectively, as shown in Fig. 3-3. With these values, we can find that the DoI value between David and Daniel is the same as that between Daniel and Denis. However, from Fig. 3-3, intuitively we know the intimacy between David and Daniel should be closer than between Daniel and Denis.

Our algorithm is based on extracting the IDF value at each facial landmark by rooting down a full family tree. With the IDF values, leaf-nodes can be compared based on their DoI values. The main contribution of this chapter is that the efficient IDF feature is proposed to replace LBF feature. This can greatly reduce the feature dimensionality, while a promising performance can still be achieved. More importantly, our algorithm runs much faster and requires less memory than that using LBF. For example, for setting with: $l = 68$, $t = 10$, and $d = 7$, the feature dimensionality of IDF is $68\times10\times1 = 680$, rather than 43,520 ($=68\times10\times64$) for LBF. In other words, the dimensionality is reduced by 64 times.

3.4 Validation Results and Comparison to LBF Feature

To validate the effectiveness, efficiency, and less memory usage of our proposed IDF method on face alignment compare to LBF [1] feature, we did intensity experiments on the public datasets.

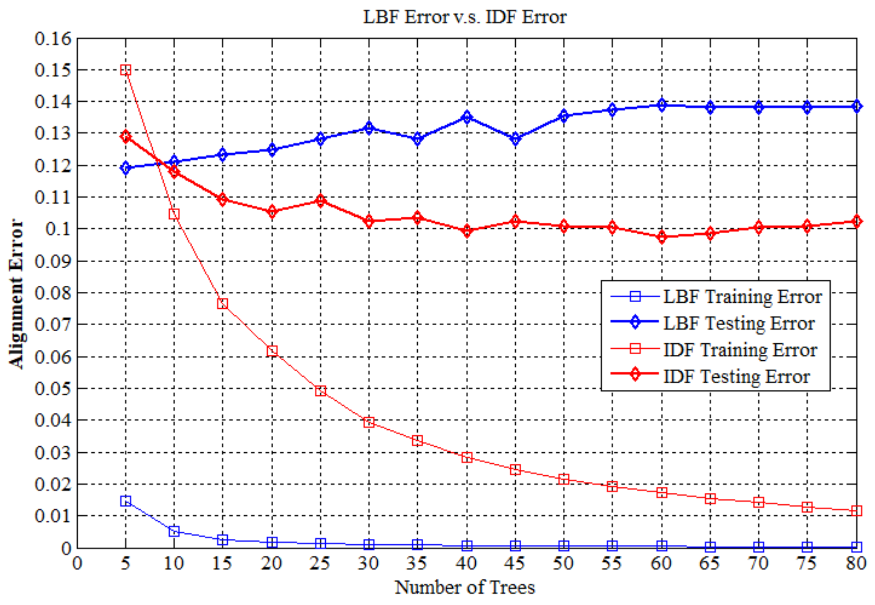


Fig. 3-6 A comparison of the alignment errors of the IDF vs LBF algorithms on the LFPW dataset [20], with tree depth = 7,training: 500, testing: 300.

To demonstrate the effectiveness of IDF for face alignment, we set tree depth, maximum number of stages, and number of landmarks at 7, and 68, respectively, and measure the respective alignment errors using LBF and IDF feature. Fig. 3-6 shows the alignment errors in the training and testing stages, based on the LFPW dataset [20], with different numbers of trees (same setting, 500 sample images are used for training and 300 for testing). From the results, we can see that our proposed IDF algorithm can achieve, on average, an error of around 0.10, when the number of trees is more than 10, while the minimum error achieved by the LBF-based algorithm is 0.12. Therefore, our algorithm can achieve an improvement of about 20%, in terms of alignment error, when compared to the LBF-based algorithm.

Another factor we should consider is the number of trees required to achieve a specific alignment error. From Fig. 3-6, we can see that using about 10 trees for our algorithm can achieve even smaller errors than that of LBF using more than 70 trees. As shown in Table 1 and Table 2, although LBF performs better in the first 3 stages, IDF can always achieve better performance at later stages, since its alignment error converges steeper than LBF. In other words, IDF converges faster in the coarse-to-fine search since it is with a higher discriminative power.

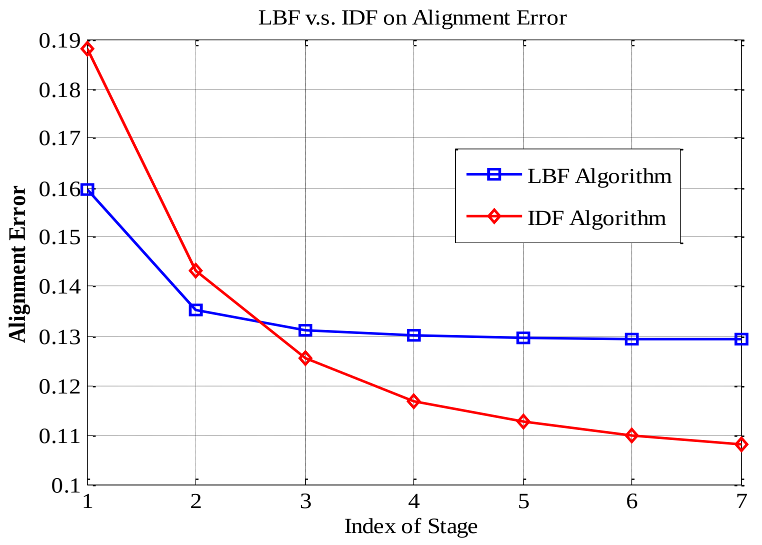
Table-3.1 Alignment errors at different stages, with different number of trees, based on the LBF algorithm.

	Number of Trees									
stage	5	10	20	30	40	50	60	70	80	*Avg.*
1	0.1765	0.1714	0.1630	0.1583	0.1583	0.1583	0.1533	0.1495	0.1485	**0.1597**
2	0.1411	0.1341	0.1410	0.1315	0.1326	0.1315	0.1397	0.1387	0.1390	**0.1352**
3	0.1386	0.1382	0.1390	0.1295	0.1293	0.1292	0.1252	0.1251	0.1276	**0.1312**
4	0.1385	0.1381	0.1389	0.1287	0.1287	0.1287	0.1240	0.1226	0.1232	**0.1301**
5	0.1384	0.1380	0.1388	0.1285	0.1285	0.1285	0.1235	0.1217	0.1209	**0.1296**
6	0.1384	0.1380	0.1388	0.1284	0.1284	0.1284	0.1234	0.1212	0.1198	**0.1294**
7	0.1384	0.1380	0.1388	0.1283	0.1283	0.1283	0.1233	0.1209	0.1193	**0.1293**

Table-3.2 Alignment errors at different stages, with different number of trees, based on the IDF algorithm.

	Number of Trees									
stage	5	10	20	30	40	50	60	70	80	*Avg.*
1	0.1924	0.1915	0.1873	0.1937	0.1886	0.1914	0.1826	0.1810	0.1856	**0.1882**
2	0.1636	0.1583	0.1472	0.1462	0.1412	0.1360	0.1312	0.1318	0.1326	**0.1431**
3	0.1540	0.1412	0.1294	0.1283	0.1206	0.1266	0.1112	0.1129	0.1136	**0.1254**
4	0.1445	0.1309	0.1188	0.1192	0.1119	0.1091	0.1041	0.1059	0.1073	**0.1168**
5	0.1380	0.1249	0.1136	0.1142	0.1076	0.1057	0.1010	0.1032	0.1049	**0.1126**
6	0.1334	0.1200	0.1114	0.1104	0.1051	0.1039	0.0990	0.1015	0.1034	**0.1098**
7	0.1291	0.1180	0.1093	0.1089	0.1036	0.1024	0.0974	0.1005	0.1025	**0.1080**

Fig. 3-7 (top) visualizes the alignment errors of the LBF and IDF methods, with different numbers of stages. We can see that the curve for IDF is much steeper than that for LBF, which means the IDF feature is more discriminative than LBF and it enjoys bigger convergence rate at later stages. An explanation to this is that the IDF value is represented with floating-point number, which has a stronger representation than LBF binary, boolean-like, values. Fig. 3-7(bottom) shows the alignment errors of IDF with a larger number of stages, the less alignment error can achieve. To make a balance between computational complexity and fitting accuracy, using 7 stages is a compromise. Therefore, in the rest of this chapter, our algorithm uses 7 stages for experiments.



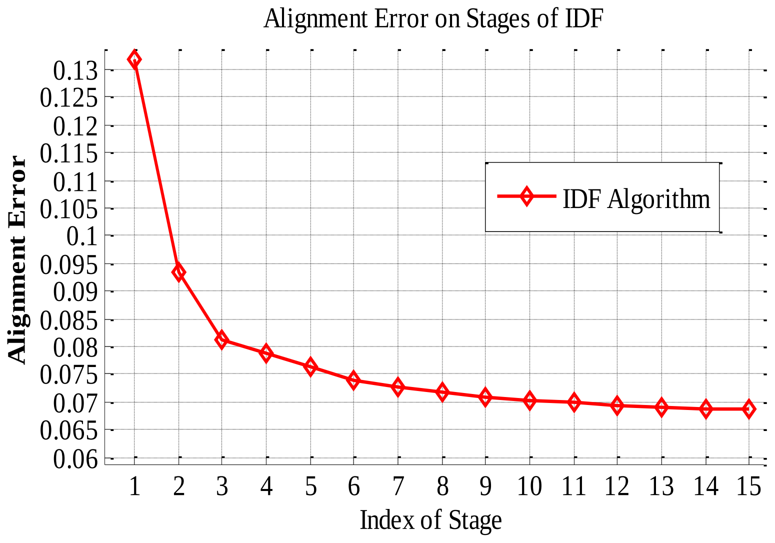


Fig. 3-7 Alignment errors with different numbers of stages: (top) LBF vs IDF, up to 7 stages, and (bottom) IDF only, up to 15 stages (tree depth = 7, LFPW dataset [20]).

Having analyzed the LBF algorithm, we found that there are two costs: (1) feature cost, and (2) regression cost, in the inference stage. The feature extraction and linear regression take up about 20% and 80% of the total computation, respectively. Since our proposed IDF is derived from the pose-indexed feature as LBF does, which means IDF, same as LBF, takes same ratio on computation. As IDF has its dimensionality an order of magnitude lower than that of LBF, the computational complexity for linear regression (the *LibLinear* package is used for both IDF and LBF) is greatly reduced compare to LBF based algorithms. As shown in Fig. 3-8, the number of frames processed per second, based on IDF, is about 2 times faster than LBF, with the same setting.

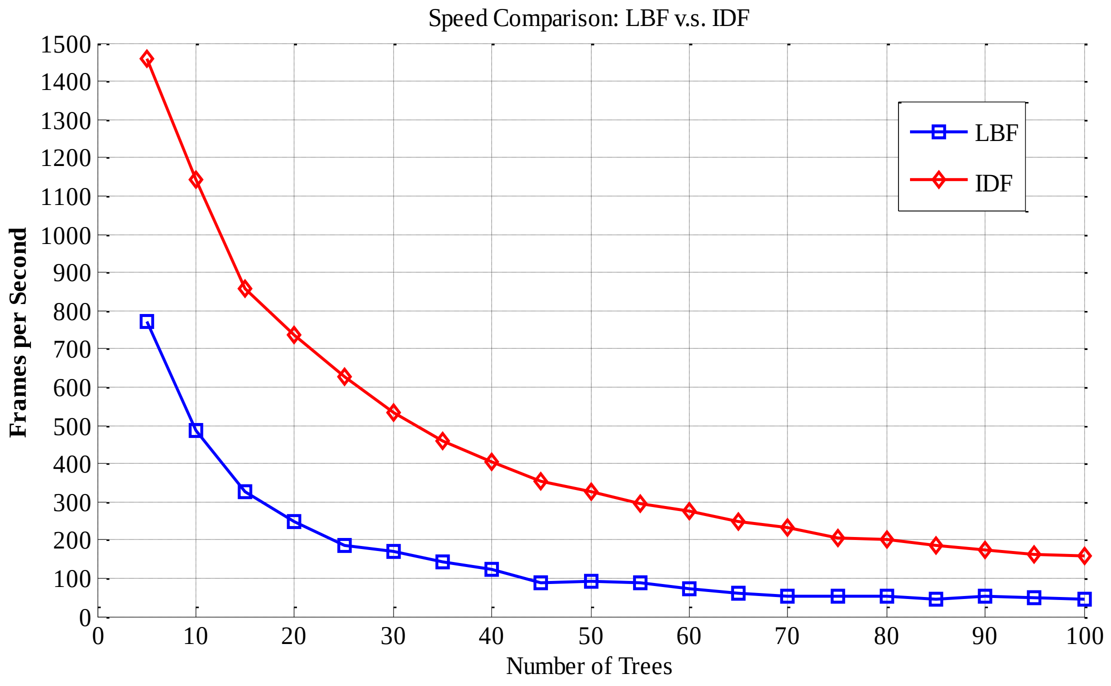


Fig. 3-8 The speed in terms of number of frames per second for the IDF vs LBF algorithms (tree depth = 7, Helen dataset).

When the tree depth increases, the feature dimensionality of LBF increases exponentially, while the IDF algorithm increases linearly. In addition to computational efficiency, memory requirement is also an important issue for real applications. Such as mobile devices, where memory bandwidth is limited, which will set a practical barrier to the algorithms with big-size models. Because of the lower dimensionality, IDF scheme enjoy less weights on the regression step. As experimental results shown in Fig. 3-9, obviously our proposed IDF feature can save an order of magnitude on memory requirement at the inference stage.

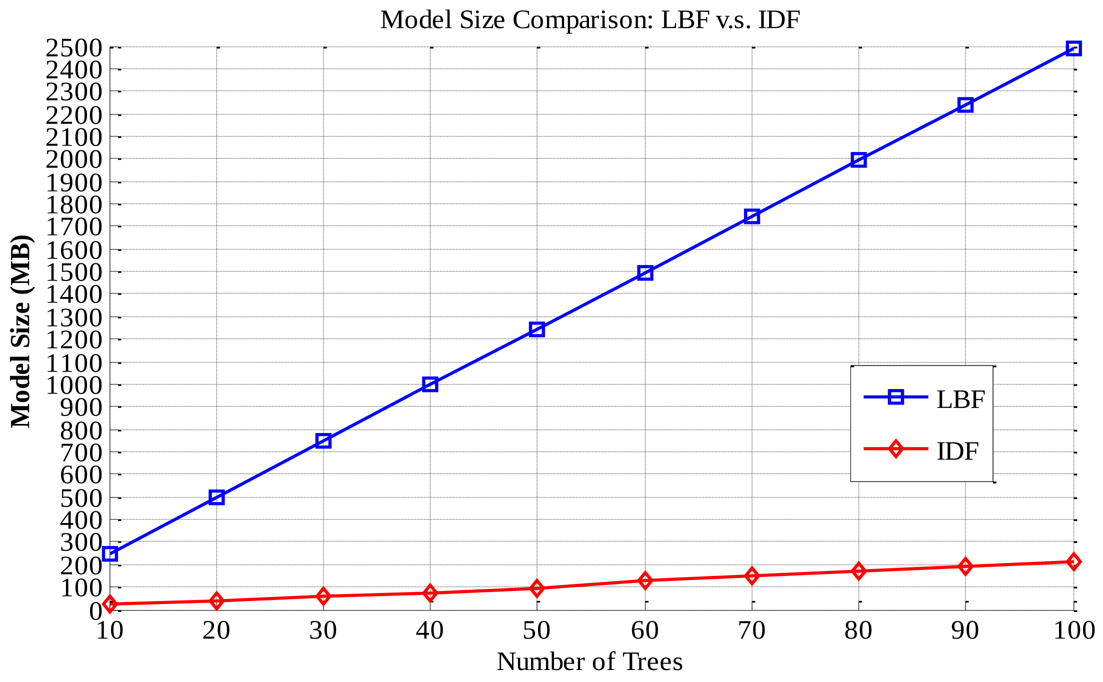


Fig. 3-9 Memory requirements (MB) of IDF vs LBF with different numbers of trees (tree depth: 7, Helen dataset) at the inference stage.

3.5 Training with Initial Shapes from Similar Samples Spanned Subspaces

Sensitive to the initial shape is the limitation of regression-based models, which means using a mean face as the initial shape, they incline to unsatisfied results on images with unseen profile faces. In [5], a conditional regression forest was proposed for face alignment, in which annotated samples are used to train a classifier to detect the face pose with discriminative features inside and outside the face-bounding boxes. Based on the annotated face poses, a few cascade regression forest models are trained, instead of a single model only. In the inference stage, once the face pose has been detected using the pose detector, the probability of the head pose is estimated from the query face image, and the corresponding trees are selected for later face alignment. In [5], a face dataset with different poses and with 10 landmark points was created. The dataset can be labelled manually, as it was in [5], so that the learning will be more precise. However, it is overlooked issue that the tedium of labeling pose faces will make the labeling results imprecise. For example, it is an ambiguous task for human eyes to decide a face with a pose with 45-degree angle from faces with 30-degree angle or 60-degree angle.

In [2], a pose detector is employed for estimating initial shapes, based on the *k* nearest neighbors selected from training samples, which uses two efficient and effective features, namely the histogram of oriented gradients (HOG) [22] and local binary patterns (LBP) [23], for searching example face images with a similar pose and texture appearance to the query face. The local appearance of feature points can be accurately approximated with locality constraints. Therefore, with the searched training faces, which have similar poses and textures to a query face, more accurate initial shape model can be constructed in the inference stage. In [2], although *k* nearest neighbors to the query face are searched with locality constraints, a relatively narrow subspace may be produced, based on the *k* training samples. What's more, this method will spoil the generalization capability of the learned model and add an additional detecting module for face alignment.

To further improve the performance, we refine the face initialization by using the k-means clustering algorithm. Different from the two above-mentioned methods [2, 5], our algorithm does not use any pose detector to consider similar texture. In our training strategy, the initial faces are selected based on the target face to span a sample subspace.

As using random initial faces in the training phase can improve the generalization capability of the alignment method, this means that the trajectory of face alignment through all regression stages, can be learnt from training samples. Intuitively, for a face with a large pose, the shape trajectory of a left-pose face cannot be learnt from a right-pose face. Therefore, initial shapes should be constraint in the subspace spanned by similar shapes, which can help to learn the pose information implicitly.

In this section, we devised a more efficient scheme for the training process. We consider 68 landmark points in face images, and we evaluate our algorithm using some standard public datasets, such as the LFPW dataset [20] (811 training + 224 testing images taken under unconstrained conditions , i.e., in the wild, with large variations in the pose, expression, illumination and with partial occlusions) and Helen dataset [21] (2000 training + 330 testing, the images exhibit a large variety in appearance, such as pose, expression, ethnicity, age and gender, as well as the general imaging and environmental conditions).

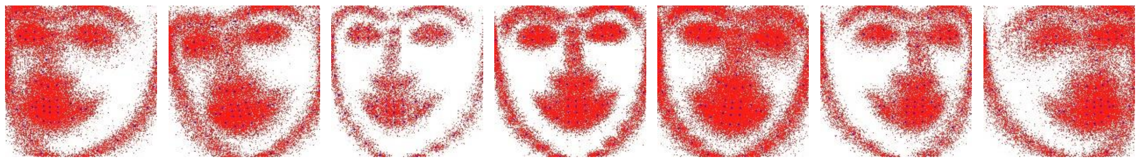


Fig. 3-10 Clustering 7 groups of different facial poses through k-means

We use the *k*-means algorithm to cluster the training samples into numbers of groups, as shown in Fig. 3-10. Then, for training each target face image, instead of blind initial faces from the whole training dataset, we choose initial faces only from the cluster with a similar pose to the target face at training stage. Therefore, the model is learned with the pose information from the spanned pose space of selected neighbor examples, which will represent the target faces well.

Experimental results in Fig. 3-11 show that the "IDF + Clustering" training scheme further improves the alignment error, when compared to the non-clustering scheme.

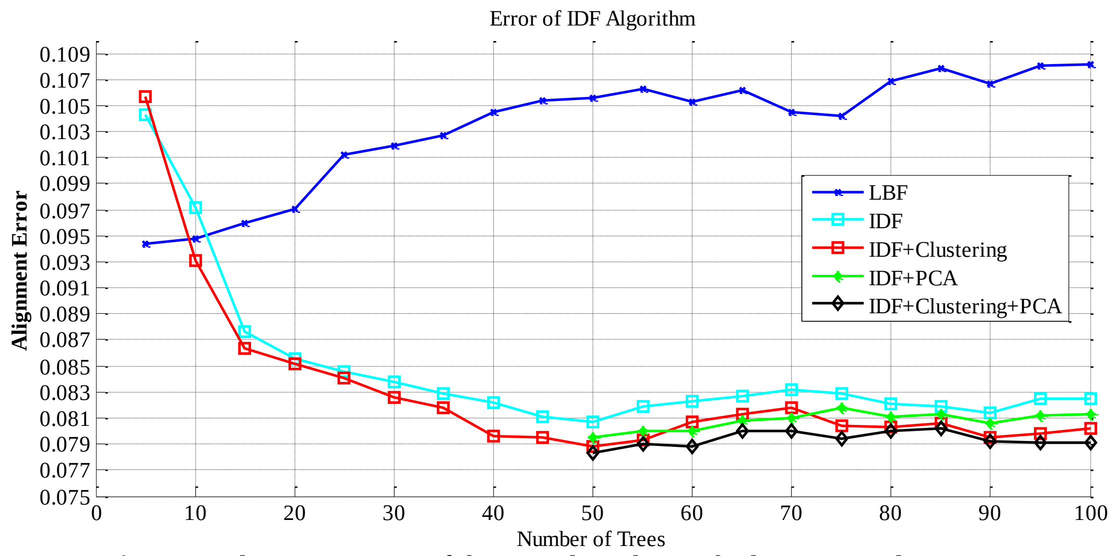


Fig. 3-11 Alignment errors of the IDF algorithm with clustering and/or PCA (tree depth: 7, stages: 5, Helen dataset [21]).

Table-3.3 Feature dimensions of IDF and IDF after using PCA (PCA* means: IDF+PCA, keeping 97% of variance).

	Number of Trees									
	10	20	30	40	50	60	70	80	90	100
IDF	680	1360	2040	2720	3400	4080	4760	5440	6120	6800
PCA*	N/A	N/A	N/A	N/A	768	806	837	852	879	888

The higher the feature dimension, the more the number of linear-regression weights for the regression model, which requires more computations and more memory on the inference stage as all the weights of the models for the cascaded stages need be loaded into memory. Another advantage of using IDF is, compared to LBF, it can apply PCA to reduce its feature dimensionality, because IDF is represented by floating-point numbers. From Table 3.3, we can see that, when the dimension becomes higher, retaining eigenvectors with 97% of variance can reduce the feature dimension by 80%~90%, and a comparable or even better performance can be achieved. Balancing the overhead cost of PCA computation and the relax on linear regression after dimension reduction, theoretically a more optimal and faster solution can be found when the feature dimension of IDF growing up. However, it is hard to do PCA compression with LBF feature's binary boolean-like values. Therefore, IDF with a higher dimensionality can be adopted to achieve both efficiency and accuracy, which is hard to LBF feature. Fig. 3-5 shows the whole workflow of proposed algorithm, and the training and fitting stages are described in **Algorithm** 3.1 and **Algorithm** 3.2, respectively.

Algorithm 3.1 IDF Training Stage:

Input: Training data (I_i, S_i, $\bar{S}_i$) for i=1, …, N, where I_i represents a face image, S_i is the corresponding shape, and N is the number of training samples.

Output: Regressors: $R = (R_1, \dots, R_T)$, T: stage number.

```
1: Using $k$-means to cluster shapes in $S$ into $K$ clusters $C = (C_1, \dots, C_K)$, randomly sample initial shapes for each target shape from its belonging cluster $\bar{S}_i \in C_i$ as source shapes;
2: for $t$=1 to $T$ do
3:     for all $i \in (1 \dots N)$ do
4:         $\Delta S_t^i = S_t^i - \bar{S}_t^i$          ⇒ Calculate shape increment: $\Delta S_t^i$
5:         $f_t^i = \phi_t(I^i, S_{t-1}^i)$          ⇒ IDF features derived from pose-indexed features
6:     end for
7:     $R_t = \arg\min_R \sum_i |R(f_t^i) - \Delta S_t^i|$   ⇒ train linear regressor $R_t$
8:     for all $i \in (1 \dots N)$ do
9:         $\bar{S}_t^i = \bar{S}_t^i + R(f_t^i)$          ⇒ update current shape
10:    end for
11: end for
```

Algorithm 3.2 IDF Fitting Stage:

Input: Testing image I, initial (mean) shape S^0, trained regressors: $R = (R_1, \dots, R_T)$, T: stages

Output: Estimated pose S^T

```
1: for $t$=1 to $T$ do
2:     $f_t = \phi_t(I, S_{t-1})$          ⇒ IDF features derived from operation: $\phi_t(I^i, S_{t-1}^i)$
3:     $\Delta S = R_t(\phi_t(I, S_{t-1}))$    ⇒ apply linear regressor $R_t$
4:     $S_t = S_{t-1} + \Delta S$          ⇒ update pose
5: end for
```

3.6 Experimental Results and Parameter Settings

We analyzed the encoding process of IDF, and found that the IDF value of each node in random forests is affected by two parameters: the *difference value d* between two brother nodes and the *magnitude value k* for each generation level. However, since the final encoded values of all the nodes are relative values, one of these two parameters can be fixed and another one used for fine-tuning. In our experiments, we fix the value of *d* to 1, and plot the alignment error curves for different values of *k*.

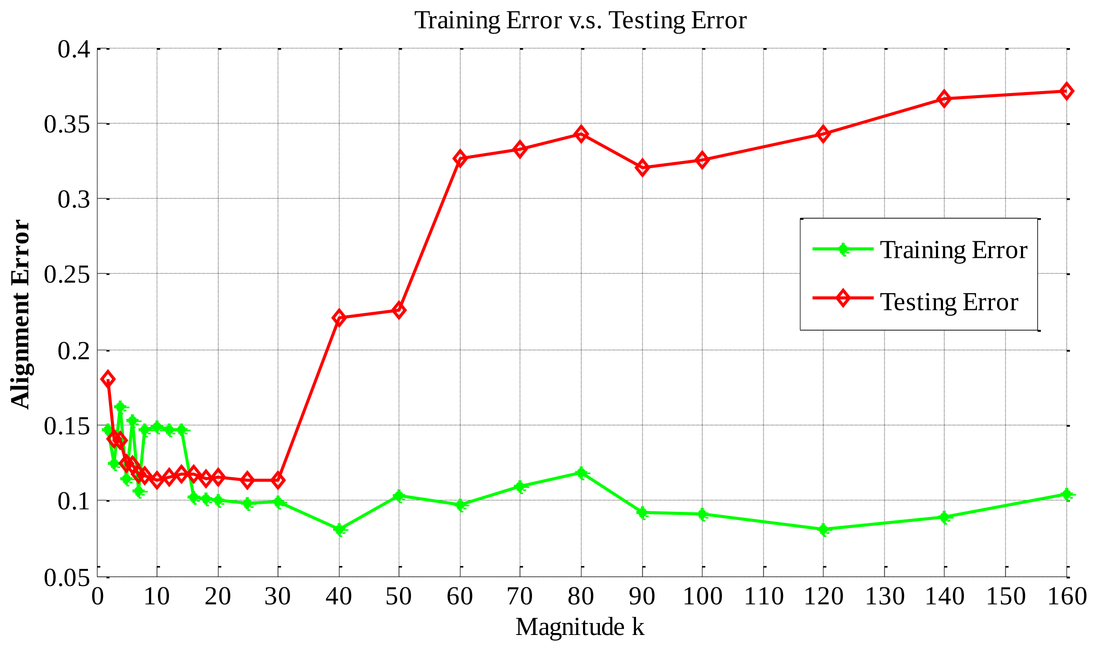


Fig. 3-12 Alignment errors of different magnitude values of *k*.

As shown in Fig. 3-12, the alignment errors become the lowest, when the *magnitude value k* is in the range from 10 to 30 (for the tree depth set at 7). This means when the *magnitude value k* is within this range, the encoded values keep the discriminative capability. Therefore, for our proposed IDF feature, the optimal setting is as follows: tree depth: 7, maximum number of stages: 7, number of trees in a forest: 11, number of initialization faces: 50, number of shape clusters: 7, and magnitude value *k*: 10. The trained model, based on our proposed IDF feature and framework, can achieve a comparable alignment quality to state-of-the-art methods [1, 6, 13, 15]. Meanwhile, our algorithm can run at a speed of more than 1,000 frames per second (FPS) on a desktop computer (Intel Core i7 4790 CPU @3.6GHz, 16GB RAM) with C++ code after thread parallelization on 8-core CPUs.

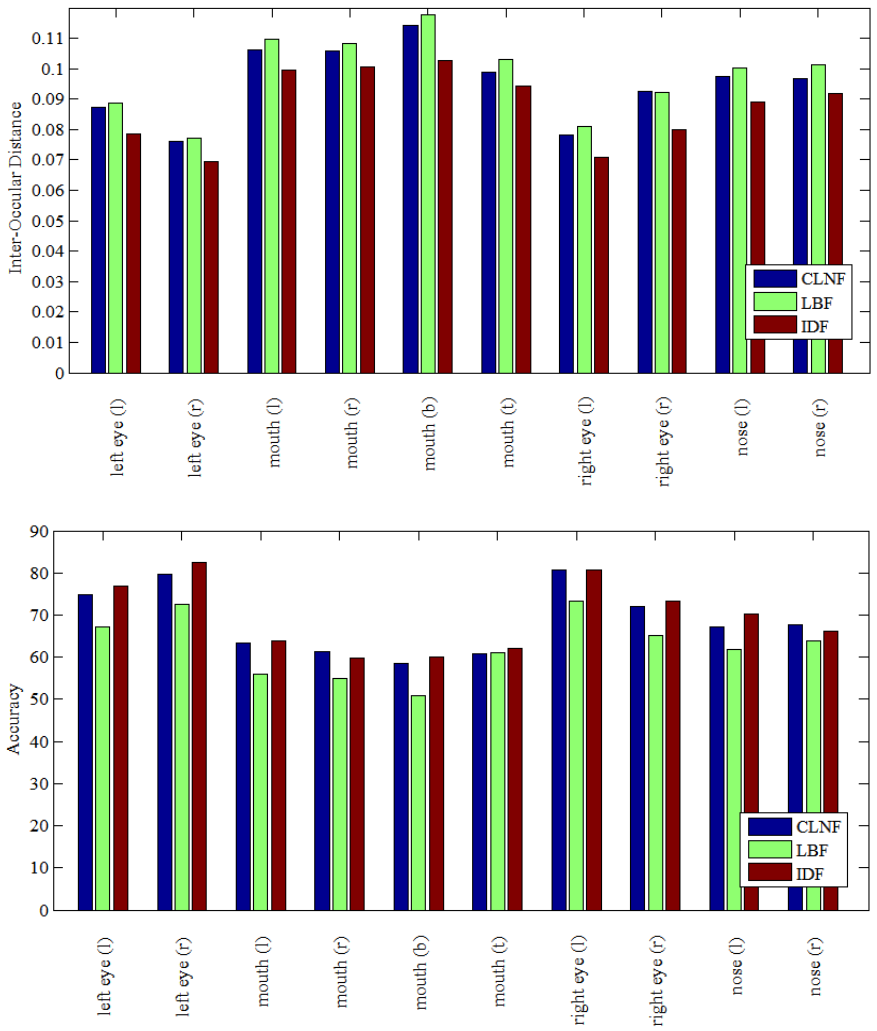


Fig. 3-13 Comparison of LBF [1], CLNF[27] and IDF, with performance on accuracy and InterOccular distance criterion on 10 facial landmark points in the Helen dataset.

The performance of IDF method, LBF [1], and CLNF [27], in terms of accuracy and the Inter-Occular distance criterion, for different facial landmarks (with 10 facial landmark points) are shown in Fig. 3-13, which demonstrates that our proposed method IDF is comparable to or, in many cases, outperformances recent state-of-the-art methods. The Fig. 3-13 also shows that, based on these two criteria, it is very clear all the methods are challenged by the facial landmark points in the mouth area, since the mouth facial landmark points represent significantly changed expression of human faces. For facial landmark points in the mouth area, our proposed method IDF has clearly improvement than same regression-based method LBF [1] method and can compare to the classic point distribution model (PDM) based method, CLNF [27] method.

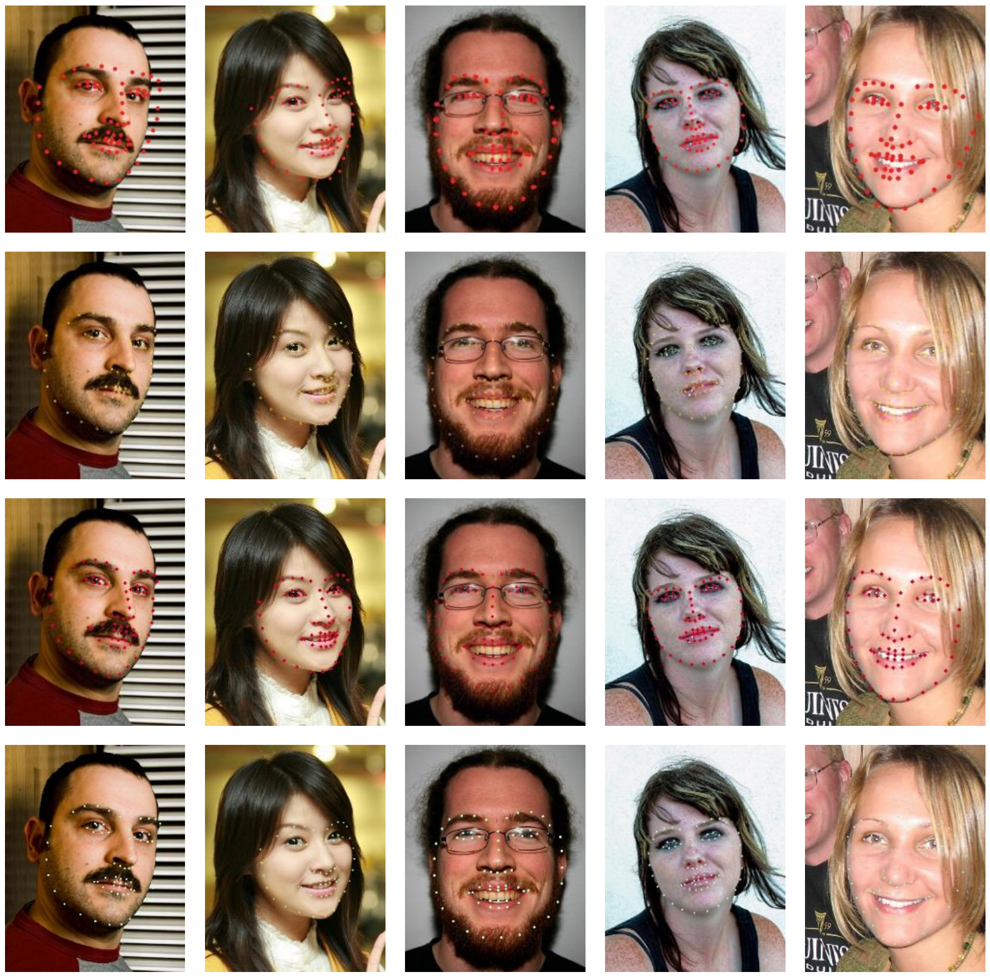


Fig. 3-14 Fitting results, with 68 landmarks, based on different methods and the Helen dataset: Row 1: LBF [1], Row 2: One-Milli-Second [15], Row 3: CLNF[27], and Row 4: IDF.

Fig. 3-14 demonstrates some results of the IDF based approach, and shows that IDF can locate landmarks accurately on faces with different poses and expressions, with occlusion, as well as faces with accessories (glasses). Our proposed method achieves promising performance, compared to the state-of-the-art algorithms [1, 15, 27]. For the linear regression setting, the *LibLinear* package [7] is used for both LBF and IDF, and the linear regression type is set as L2R_L2LOSS_SVR, i.e., L^2-regularized L^2-loss support vector regression (primal), in which Newton method with trust region step control is employed [30].

3.7 Conclusions

In this chapter, we have proposed a novel, simple but effective and discriminative feature, and explored the random-forest based cascaded regression model. The core insight of our proposed intimacy definition feature (IDF) is we elaborately construct a full binary family tree for computing any two members' degree of intimacy (DoI), which can encode the path from the root to a leaf-node as a floating-point number.

The contributions of the method are threefold. Firstly, compared to local binary feature (LBF) which produces a sparse binary vector from each tree, IDF yields a scalar value. IDF helps the regression-based model achieve state-of-the-art performance, in terms of alignment accuracy and computational efficiency, and memory requirement. Secondly, we have addressed the fact that regression-based approaches are sensitive to shape initialization. Rather than using a few blind initializations, we choose initial shapes from their similar samples spanned subspaces at training stage. With this initialization strategy, the cascaded regression approach is capable of learning more accurate alignment trajectory and further improving the generalization capability of the trained forests. Finally, since IDF is a generic random-forest based feature which can be applied to other computer vision tasks, the IDF feature has enriched random-forest based research topics.

Until now, real-time face alignment is still a challenging task. Although lots of researchers have put efforts into this research area and numerous algorithms have been proposed, a highly robust and efficient algorithm is still on the way. Limited by the capacity of pixel-based features, the derived IDF feature is susceptible to image noise, compared to manually crafted feature, e.g., the SIFT feature, so further investigation is necessary to tackle these problems for faces with noise, large poses and occlusion.

Chapter 4. Efficient Likelihood Bayesian Constrained Local Model

The constrained local model (CLM) proposes a paradigm that the locations of a set of local landmark detectors are constrained to lie in a subspace, spanned by a shape point distribution model (PDM). Fitting the model to an object involves two steps. A response map, which represents the likelihood of locations for a landmark, is first computed for each landmark using local-texture detectors. Then, an optimal PDM is determined by jointly maximizing all the response maps simultaneously, with a global-shape constraint. This global optimization can be considered a Bayesian inference problem, where the posterior distribution of the shape parameters, as well as the pose parameters, can be inferred using maximum a posteriori (MAP). In this chapter, based on the CLM model, we present a novel CLM variant, which employs random-forest regressors to estimate the location of each landmark, as a likelihood term, efficiently. This novel CLM framework is called efficient likelihood Bayesian constrained local model (elBCLM). Furthermore, in each stage of the regressors, the PDM local non-rigid parameters, i.e. the shape parameters, of the previous stage can work as shape clues for training the regressors for the current stage. To further improve the efficiency, we also propose a feature-switching scheme used in the cascaded framework. Experimental results on benchmark datasets show our approach achieves about 3 to 5 times speed-up, when compared with the existing CLM models, and improves by around 10% on fitting accuracy, when compared with the other regression-based models.

4.1 Introduction

The main goal of face alignment is to locate the semantic structural facial landmarks, such as the eyebrows, eyes, nose, mouth, and face contour, accurately, as illustrated in Fig. 4-1. This information about facial landmarks is crucial for understanding and analyzing face-related research and applications, such as expression recognition [9], face recognition [10], and face hallucination [11].

The classic active appearance model (AAM) [4] and constrained local models (CLM) [28, 27] for face alignment, which are based on Newton's gradient-descent methods, use facial-appearance information (i.e. the image pixel-intensity patterns in the area around the landmarks), and face-shape information (i.e. the face shape defined by the landmark coordinates) to locate facial landmarks. The shape model, used in active contour model (ASM) [3], AAM [4] and CLM, is usually described by finding the parameters of a statistical shape-distribution model, i.e., PDM. PDM is a linear model, where facial shapes are modeled as a linear combination of the eigen-shapes around the mean shape, which can be generalized to represent unseen facial shapes.

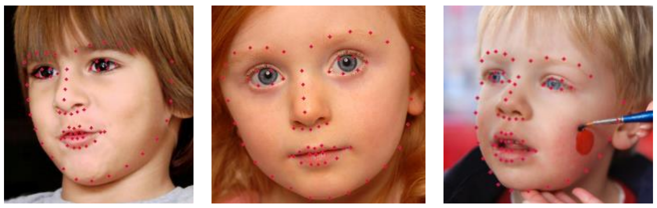


Fig. 4-1. Facial landmarks fitting located by elBCLM (the Helen dataset [7]).

Recently, a new family of face-alignment algorithms has emerged [1, 6, 51, 8, 13], which directly learns regressors from image-feature descriptors to the target shape increment. These regression-based methods are gaining popularity, due to their excellent performance and high efficiency on face alignment. The regression-based methods do not explicitly learn any shape model, rather, they learn models directly from facial appearance or manually designed features, and then predict landmark locations based on trained models.

In this chapter, we propose a novel framework, which takes the advantages of both regression-based methods and CLM models. In our framework, the conventional local-response maps, which are computed by using random-forest-based regressors, can be computed efficiently and cover a large area for predicting the landmark locations. Our method can also be considered a regression-based framework, with a PDM shape constraint. Therefore, we devise this algorithm as an efficient likelihood Bayesian constrained local model (elBCLM). With the PDM constraint, regression-based methods [51] with same setting can achieve 5% improvement, in terms of accuracy on Helen dataset (2000 training+330 testing, exhibiting a large variety in appearance as well as general imaging and environment). Furthermore, for each stage of the regressors, the PDM local non-rigid parameters from the previous stage can be taken as a shape clue on training each regression model, which enables the elBCLM scheme get a further 5% improvement, i.e., totally 10% gain.

The remainder of the chapter is organized as follows. In Section 2, an overview of existing, related methods for face alignment is given. In Section 3, CLM is presented, and our proposed model, elBCLM, is described in detail in Section 4. A feature-switching strategy to balance the computation and efficiency in a cascaded framework, is proposed in Section 5. Experiment results are given and discussed in Section 6, and a conclusion is given in Section 7.

4.2 Previous Works

Recent face alignment approaches can be classified into two major categories, the classic Newton's gradient-descent-based methods and the regression-based methods. The Newton's gradient-descent-based methods are used in the active appearance model (AAM) [4], which learns the holistic model parameters by updating the Jacobian and Hessian matrices in the fitting stage, and the constrained local model (CLM) [52, 28, 27], which determines each landmark independently, by using local-appearance information and learning local-patch response maps. The CLM models also embed the face-shape model, i.e. the point distribution model (PDM) as the shape constraint. The PDM is a linear model with parameters to represent the shapes. It is used to estimate the likelihood of the landmark potential locations, which is important for model fitting, as it can act as a prior in the Bayesian framework.

Recently, regression-based models [1, 6, 13, 15] have shown significantly better performances than the AAM and CLM frameworks. These models do not learn any shape model explicitly, but only learn models for predicting the landmarks directly from facial-appearance feature patterns. The supervised descent method (SDM) [6] is the pioneer work on regression methods. SDM formulates the face-alignment task as a general optimization problem, which is approximately solved by learning successive mapping functions from local-appearance feature patterns, such as histogram of oriented gradients (*HOG*) or scale-invariant feature transform (SIFT) features, to the shape updates using linear-regression models. In [1], the authors presented highly efficient local binary features (LBF), which are derived from local pixel shape-indexed features, with a linear-regression framework to achieve faster speed with comparable quality. In [51], a local lightweight feature, namely intimacy definition feature (IDF), was proposed. This feature can achieve about twice the speed-up and more than 20% improvement in terms of alignment error, when compared to the LBF [1] method.

The regression-based methods take advantage of shape information in a limited sense, such that all the points are updated jointly, i.e. each point is described by linear regression with features from all other points. Thus, the shape-pattern constraint is implicitly embedded in the model. Since the shape prior is not employed explicitly, current regression-based methods have the inherent limitation of ignorance or ineffective usage of the shape information, which is the main issue this chapter will tackle.

4.3 Constrained Local Models

4.3.1 The Shape Model - PDM Model

Generally, the shape $\boldsymbol{X}$ of a point distribution model (PDM) is represented by the 2D vertex locations of a mesh, with a $2n$ dimensional vector: $\boldsymbol{X} = (x_1, y_1, .., x_n, y_n)^T$. Traditional way of building a PDM requires a set of shape-annotated images that have been aligned in scale, rotation, and translation by the Procrustes analysis. Applying principal component analysis (PCA) to a set of aligned training examples, the shape can be expressed by a linear parametric model, as follows:

$$\boldsymbol{X}_i = s\mathbf{R}(\overline{\boldsymbol{X}}_i + \boldsymbol{\Phi}_i\mathbf{q}) + \boldsymbol{t}, \tag{4.1}$$

where $\boldsymbol{X}_i$ denotes the 2D landmark locations of the PDM's i^{th} landmark, and **p** = {s, **R**, **t**, **q**} denotes the PDM parameters, which consist of a global scaling s, a rotation **R**, a translation **t**, and a set of non-rigid parameters **q**. Here, $\overline{\boldsymbol{X}}_{\boldsymbol{i}}$ denotes the mean location of the i^{th} PDM landmark in the reference frame (i.e. $\overline{\boldsymbol{X}}_{\boldsymbol{i}} = [\bar{x}_i, \bar{y}_i]$ for a 2D model), and $\boldsymbol{\Phi}$ is the shape-subspace matrix formed by the leading n eigenvectors (retaining a userdefined variance, e.g. 99%). Therefore, $\mathbf{q}$ can be regarded as a vector of shape parameters. From the probabilistic point of view, the non-rigid shape parameters $\mathbf{q}$ show a Gaussian distribution, leading to the following prior:

$$p(\mathbf{q}) \propto \mathcal{N}(\mathbf{q}; \mathbf{0}, \mathbf{\Lambda}), \ \ \mathbf{\Lambda} = \mathrm{diag}\{[\lambda_1; \lambda_2; \dots; \lambda_\mathrm{m}]\} \tag{4.2}$$

where λ_i denotes the eigenvalue of the i^{th} mode of deformation. $\mathbf{\Lambda}$ is constructed from the training set, based on how much shape variation in the training is explained by the i^{th} parameter, with λ_i corresponding to the $\mathbf{q}_i$ parameter.

The CLM models have attracted some interest, as they solve many of the drawbacks of holistic approaches. A CLM model normally consists of two parts: a statistical shape model and patch experts (also called local detectors). Both the shape model and patch experts can be trained offline, and then used for online landmark detection, which is achieved by fitting the CLM to a given image. The deformable model is controlled by the parameters in $\mathbf{p}$, and the instance of a model can be described by the locations of its feature points $\boldsymbol{X}_i$ in an image $\boldsymbol{I}$, as illustrated in Fig. 4-2.

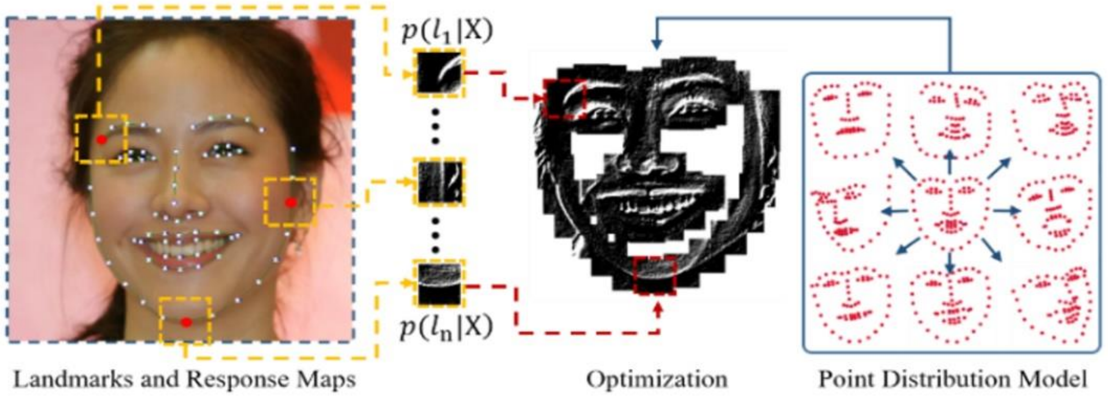


Fig. 4-2 Overview of the CLM fitting process.

4.3.2 CLM Model and Fitting Process

The CLM fitting is generally composed of searching the PDM parameters **p**, which jointly minimizes misalignment error over all the landmarks, regularized properly, as follows:

$$\mathcal{E}(\mathbf{p}) = R(\mathbf{p}) + \sum_{i=1}^{n} D_i(\boldsymbol{X}_i; I), \tag{4.3}$$

where R penalizes the complex deformations (i.e. the regularization term) and D_i denotes the measure of misalignment for the i^{th} landmark $\boldsymbol{X}_i$ in the image $\boldsymbol{I}$ (i.e. the data term). The form of regularization, which describes plausible object shapes, is related to the assumed distribution of the PDM parameters.

Two steps of CLM fitting process in probability thinking:

(1) an exhaustive local search for feature locations to generate the response maps:

$$\{p(l_i = aligned|\boldsymbol{I}, \boldsymbol{X})\}_{i=1}^{n}, \tag{4.4}$$

(2) an optimization strategy to maximize the responses of the PDM constrained landmarks.

Most innovations made to the CLM model are to replace the distribution of landmark locations, obtained from each patch-based local detector, with a simpler and more accurate predictor. For the optimization step, once the response map for each landmark have been computed, by assuming conditional independence, optimization can proceed by maximizing the following function:

$$p(\{l_i = aligned\}_{i=1}^{n}|p) = \prod_{i=1}^{n} p(l_i = aligned|\boldsymbol{X}_i) \tag{4.5}$$

with respect to the PDM parameters **p**, where $\boldsymbol{X}_i$ is parameterized as in Eqn. (4.1), and dependence on the image $\boldsymbol{I}$ is dropped for succinctness. It should be noted that some forms of CLMs use Eqn. (4.4) as minimizing the total local energy responses. In Eqn.

(4.5), l_i is a discrete random variable, denoting whether the i^{th} landmark is correctly aligned or not.

4.3.3 CLM in Bayesian Formulation

The objective of Eqn. (4.5) can be interpreted as maximizing the likelihood of the model parameters, such that all its landmarks are aligned with the corresponding locations of the object in an image. The specific form of the objective implicitly assumes conditional independence between the detected landmarks. The probability of correct alignment can be expressed as follows:

$$p(\mathbf{p}|\{l_i = 1\}_{i=1}^{n}, \boldsymbol{I}) \propto p(\mathbf{p}) \prod_{i=1}^{n} p(l_i = 1|\boldsymbol{X}_i, \boldsymbol{I}) \tag{4.6}$$

i.e. $\ln\{p(\mathbf{p}|\{l_i = 1\}_{i=1}^{n}, \boldsymbol{I})\} \propto \ln\{p(\mathbf{p})\} +$

$$\sum_{i=1}^{n} \mathbf{ln}\{p(l_i = 1|\boldsymbol{X}_i, \boldsymbol{I})\}. \tag{4.7}$$

Based on Eqn. (4.2), we have

$$\mathcal{E}(\mathbf{p}) = -\ln\{p(\mathbf{p})\} \tag{4.8}$$

$$D_i(\boldsymbol{X}_i; I) = -\mathbf{ln}\{p(l_i = 1|\boldsymbol{X}_i, \boldsymbol{I}). \tag{4.9}$$

If a non-informative (uniform) prior over the PDM parameters is assumed, the formulation in Eqn. (4.6) leads to a maximum likelihood (ML) estimate, otherwise it leads to a maximum a posterior (MAP) estimate.

The method first finds the location within each response map for which the maximum is attained, and the locations of the n landmarks are denoted as $\boldsymbol{\mu} = [\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_n]^T$. The objective of the optimization procedure is then to minimize the weighted least squares difference between the PDM and the coordinates of the peak responses in the map window, regularized appropriately as follows:

$$\mathcal{E}(\mathbf{p}) = ||\mathbf{q}||^{\mathbf{2}}_{\mathbf{\Lambda^{-1}}} + \sum_{i=1}^{n} w_i ||\boldsymbol{X}_i - \boldsymbol{\mu}_i||^2, \tag{4.10}$$

where the weights $\{w_i\}_{i=1}^{n}$ reflect the confidence on the landmark locations, making it more resistant towards partial occlusion, where occluded landmarks will be more weakly weighted. Eqn. (4.10) is iteratively minimized by taking the first order Taylor expansion of the PDM's landmarks:

$$\boldsymbol{X}_i \approx X_i^c + \mathbf{J}_i \boldsymbol{\Delta}\mathbf{p} \tag{4.11}$$

and then the parameter update is solved as follows:

$$\boldsymbol{\Delta}\mathbf{p} = -\mathbf{H}^{-1}\boldsymbol{X}_i(\boldsymbol{\Lambda}^{-1}\mathbf{p} + \textstyle\sum_{i=1}^{n} w_i \mathbf{J}_i (X_i^c - \boldsymbol{\mu}_i)), \tag{4.12}$$

which updates the current parameters: $\mathbf{p} \leftarrow \mathbf{p} + \boldsymbol{\Delta}\mathbf{p}$. Here, $\boldsymbol{\Lambda} = \boldsymbol{diag}\{[\boldsymbol{\lambda_1}, \boldsymbol{\lambda_2}, \ldots, \boldsymbol{\lambda_m}]\}$, $\mathbf{J} = [\mathbf{J_1}, \mathbf{J_2}, \ldots, \mathbf{J_n}]$ is the PDM's Jacobian matrix, $\boldsymbol{X^c} = [X_1^c; X_2^c; \ldots; X_n^c]$ is the current shape estimate, and

$$\mathbf{H} = \boldsymbol{\Lambda}^{-1} + \textstyle\sum_{i=1}^{n} w_i \mathbf{J}_i^{\boldsymbol{T}} \mathbf{J}_i \tag{4.13}$$

is the Gauss-Newton Hessian matrix. The Jacobian matrix contains the partial derivatives of the *n* landmarks with respect to the PDM parameters. i.e. the 4 global rigid parameters (s, θ, t_x, t_y) and the *m* local non-rigid parameters in **q**, where θ is the angle of rotation matrix **R**.

4.4 Proposed Model

In recent years, random forests [14] have emerged as an effective approach to learning classifiers, for a large variety of computer-vision tasks. This method is relatively simple and has many merits that make it particularly interesting for computer-vision tasks [51].

4.4.1 Random-Forest-based Cascaded Shape Regression

Many face-alignment methods work under a cascaded framework, where an ensemble of *N* regressors operates in a stage-by-stage manner, which are referred to as stage regressors. This approach was first explored in [8]. At the fitting stage, the input to the regressor R_t, at stage t, is a tuple $(\boldsymbol{I}, \boldsymbol{X}_{t-1})$, where $\boldsymbol{I}$ is an image and $\boldsymbol{X}_{t-1}$ is the shape estimated from the previous stage (the initial shape $\boldsymbol{X}_0$ is typically the mean shape of the training set). The regressors work on features, i.e. information about the current shape estimate, or other features with respect to the current shape estimate, and predict a vector of shape increment as follows:

$$\boldsymbol{X}_t = \boldsymbol{X}_{t-1} + \boldsymbol{R}_t\big(\boldsymbol{\phi}_t(I, \boldsymbol{X}_{t-1})\big), \tag{4.14}$$

where $\boldsymbol{\phi}_t(\boldsymbol{I}, \boldsymbol{X}_{t-1})$ can be referred to as the shape-indexed features, or features, which are derived from shape-indexed features, such as LBF [1] and IDF [51]. The cascade progressively infers the shape in a coarse-to-fine manner. The early regressors handle large variations in shape, while the later ones perform small refinements. After each stage, the shape estimate resembles the true shape closer and closer.

In our proposed algorithm, the feature-mapping function $\boldsymbol{\phi}_t(\boldsymbol{I}, \boldsymbol{X}_{t-1})$ generates local IDF features, which are derived from the shape-indexed feature or HOG features at the estimated landmark positions at later stages. With the assumption, proved by intensive experimental results, that the shape increments have close correlation with the local features of the landmarks, which define the face shape, given the features and the target shape increments $\{\Delta\boldsymbol{X}_t = \boldsymbol{X}_t - \boldsymbol{X}_{t-1}\}$, a linear projection matrix $\boldsymbol{R}_t$ can be learned. Most regression models [1, 6, 51, 8] share a similar framework.

4.4.2 Motivation and Proposed Method

If the weights are all the same and only one iteration is performed, then Eqn. (4.12) can be simplified as follows:

$$\Delta\mathbf{p} = -\mathbf{H}^{-1}\mathbf{X}(\boldsymbol{\Lambda}^{-1}\mathbf{p} + \mathbf{J}\boldsymbol{v}), \tag{4.15}$$

where $\boldsymbol{v}$ is the shape shift vector obtained by any response-map algorithm. Recently, researchers have worked for using different response maps [27]. Directly obtaining the shape shift vector $\boldsymbol{v}$ can overcome the limitation of local response-map filters based CLM algorithms, which can obtain the shift vector by setting $\boldsymbol{v} = \Delta\mathbf{X}$, while the accuracy can be improved by iterations from the cascaded framework:

$$\Delta\mathbf{p} = -\mathbf{H}^{-1}\mathbf{X}(\boldsymbol{\Lambda}^{-1}\mathbf{p} + \mathbf{J}\Delta\mathbf{X}). \tag{4.16}$$

This equation interprets the main idea of our proposed method, in which random-forest-based regressors are trained to obtain $\Delta\mathbf{X}$ efficiently. $\Delta\mathbf{p}$ is updated for refining the current fitted shape with the PDM model constraint. Since the response-map filters mainly work as a convolutional filter, the drawback of convolutional filters significantly hinders the CLM algorithm. If the size of a patch-based response-map window is set too small, the response-map filter cannot cover a sufficiently large area to handle large posed faces. On the other hand, if it is set too big, the computation requirement may greatly reduce algorithm's efficiency. The PDM-based prior term, as in Eqn. (4.2), can be approximated as follows:

$$p(\mathbf{q}_k|\mathbf{q}_{k-1}) \propto \mathcal{N}\left(\mathbf{q}_k|\mu_q, \boldsymbol{\Sigma}_q\right), \tag{4.17}$$

where $\mu_q = \mathbf{q}_{k-1}$ and $\boldsymbol{\Sigma}_q = \boldsymbol{\Lambda}$. This form of prior assump-tion can be greatly improved in a cascaded framework.

Inspired by this analysis, we propose a novel CLM variant, by replacing the response-map filters with regressors to find the shift vector $\Delta\mathbf{X}$. The refinement of the shift vector $\Delta\mathbf{X}$ is implicitly realized in the cascade framework. For CLM, random-forest-based regressors replace the local response maps, so likelihood can be calculated efficiently under the Bayesian framework. Because of these, we name our proposed algorithm as an efficient likelihood Bayesian constrained local model (elBCLM).

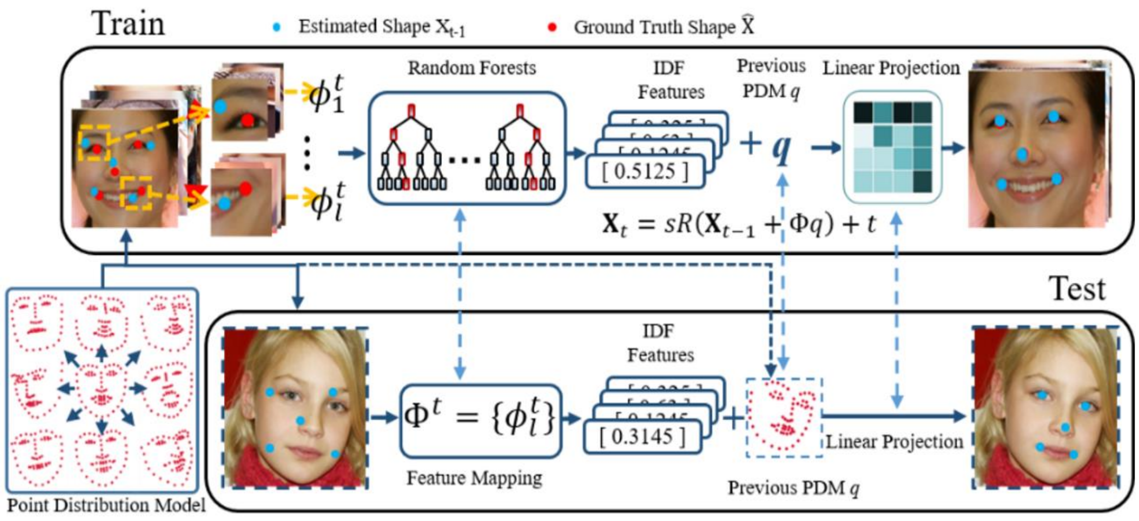
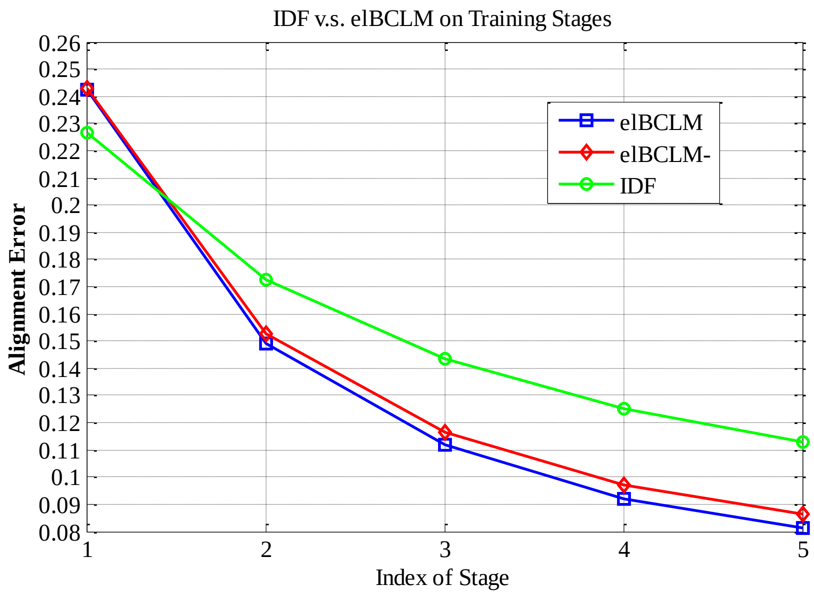


Fig. 4-3: An overview of the workflow for elBCM cascaded regression face alignment

Our proposed algorithm, using regressors to replace local response maps has the following advantages: (1) This method can circumvent the hypothesis that the local detectors are assumed conditionally independent, (2) Our method is able to avoid local optimal location, which may be caused by typical, local noise and ambiguities, since small image patches often contain limited structure, (3) This method is capable of extending the response-map with more efficient methods. The whole workflow of our elBCLM algorithm is described in Fig. 4-3, and the detail of random-forest-based regression may refer to [1, 51].

4.4.3 PDM as Prior for Regression Features

The PDM is a linear model, which parametrizes a class of shapes. It can also be used to estimate the likelihood of landmark location, given a set of feature points. This is important for model fitting, as it can act as a prior,which works as a guiding feature in the cascade framework.



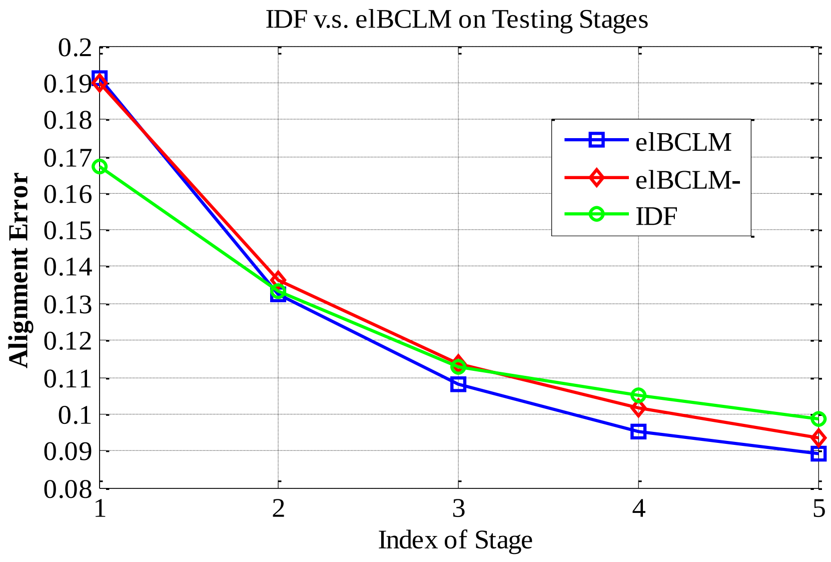
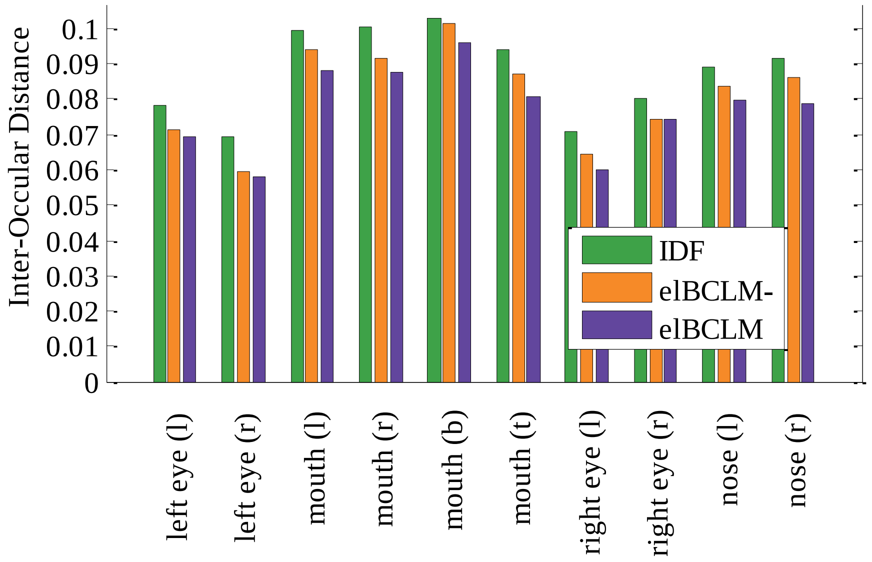


Fig. 4-4: Comparison of regression-based method IDF [5], elBCLM with PDM shape constraint only(elBCLM-), and elBCLM.

As the local non-rigid parameters **q** in the PDM **p** = {*s*, **R**, **t**, **q**} have a closer relationship to the consecutive fitting shapes, so the previous local parameters **q** can work as a guiding feature for fitting the shape at each stage. In our experiments, for the 68 facial landmarks in a face, the dimension of **q** is 136 (68×2), which can be reduced to around a dimension of 32 by using PCA, with 99% of energy retained. Therefore, in our algorithm, the dimension of **q** is set at 32. As **q** is used in regression, this means that the features in $\boldsymbol{\phi}_t(\boldsymbol{I}, \boldsymbol{X}_{t-1})$, from Eqn. (4.14), are refined.



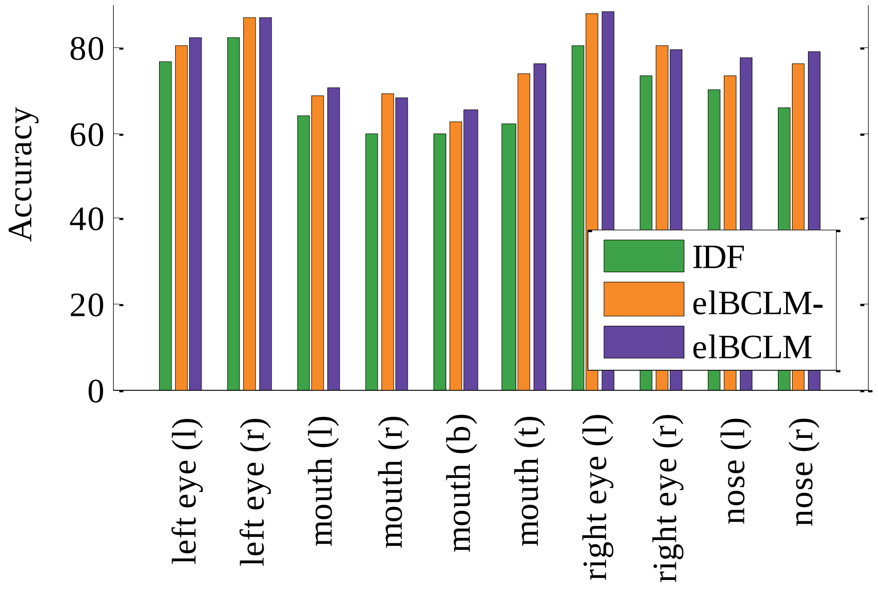


Fig. 4-5: Comparison of regression-based method IDF[51], elBCLM- (with PDM shape constraint only), and elBCLM, for 10 facial landmarks.

Fig. 4-4 shows that, both the training and fitting stages, the regression algorithm with PDM shape constraint only (denoted as elBCLM- in Fig. 4-4) can reduce the alignment error by more than 5%, when compared to IDF [51]. When the PDM's local parameters **q** are also used as a guiding feature for training the linear-regression models in each stage, our algorithm, elBCLM, is able to achieve an additional 5% improvement in terms of alignment error in the fitting stage. The performances of IDF, elBCLM- (with PDM constraint only), and elBCLM, in terms of accuracy and the Inter-Occular distance criterion, for different facial landmarks are shown in Fig. 4-5, which demonstrates that the three methods have similar relative performances for different landmarks.

4.5 Feature-Switching Scheme

In the cascade framework, the performance of the regressors can be fine-tuned. For example, adaptive window size is used, where the window size decreases when moving along the stages to balance the computation [1, 6, 51]. In [54], to handle large variations, global regression is first employed, then part regression, and finally local regression. From experiment results, we can find all features share similar trend, which means convergence curve decrease relatively faster in the earlier stages while slower in the later stages, as can be seen in Fig. 4-6. After the earlier stages, the convergence is insensitive to the features being used. Therefore, to balance speed and accuracy, a simple feature is first used, then after a certain number of stages, it is switched to a more complex, discriminative feature in later stages. We call this a feature-switching scheme. In Fig. 4-6, we show the performance of the feature-switching scheme with 6 stages. In the first 4 stages, the IDF feature is used to achieve a faster speed, and it is switched to the HOG feature 2 stages to achieve higher accuracy. The computation requirement of one HOG stage is higher than four IDF stages.

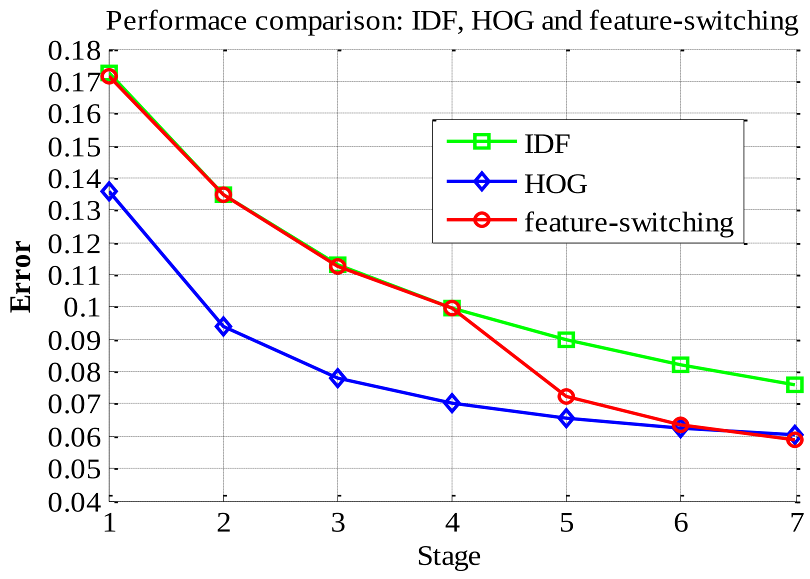


Fig. 4-6: Comparison: IDF, HOG and feature-switching(Helen dataset [21]).

4.6 Workflow and Experimental Results

4.6.1 Algorithm Workflow

The two stages of elBCLM are described in Algorithm 4.1 and 4.2, respectively.

Algorithm 4.1: elBCLM Training Stage:

Input: PDM $(\boldsymbol{X}, \boldsymbol{\Phi})$ model, training data$(\boldsymbol{I}_i, \boldsymbol{X}_i, \overline{\boldsymbol{X}}_i)$, for *i*=1, …, *N*, where ***I*** are face images, and ***X*** are shapes; *N* is the number of samples.

Output: regressors: $\boldsymbol{R} = (R_1, \dots, R_T)$, *T*: stage count.

```
1:  for $t$=1 to $T$ do
2:      for all $i \in (1 \dots N)$ do
3:          $\Delta\boldsymbol{X}_t^i = \boldsymbol{X}_t^i - \overline{\boldsymbol{X}}_t^i$          ⇒ calculate $\Delta\boldsymbol{X}_t^i$
4:          $f_t^i = \phi_t(\boldsymbol{I}^i, S_{t-1}^i)$          ⇒ IDF features + PMD's $\mathbf{q}$
5:      end for
6:      $R_t = \arg min_R \sum_i |R(f_t^i) - \Delta\boldsymbol{X}_t^i|$
7:      for all $i \in (1 \dots N)$ do
8:          $\overline{\boldsymbol{X}}_t^i = \overline{\boldsymbol{X}}_t^i + R(f_t^i)$          ⇒ update shape {Eqn. (4.14)}
9:          update $\Delta$p with $\Delta S_t^i$          ⇒ {Eqn. (4.16)}
10:         $\overline{\boldsymbol{X}}_t^i \approx \overline{\boldsymbol{X}}_t^i + \mathbf{J}_t\Delta\mathbf{p}$          ⇒ update shape w.r.t: Eqn. (4.10)
11:     end for
12: end for
```

Algorithm 4.2: elBCLM Fitting Stage:

```
Input: PDM ($\boldsymbol{X}, \boldsymbol{\Phi}$) model, testing image $\boldsymbol{I}$, initial (mean) shape $\boldsymbol{X}^0$, trained
regressors:$\boldsymbol{R} = (\boldsymbol{R}_1, \dots, \boldsymbol{R}_T)$
Output: Estimated pose $\boldsymbol{X}^T$
1: for t=1 to T do
2:     $f_t = \phi_t(\boldsymbol{I}, \boldsymbol{X}_{t-1})$            ⇒ IDF features + PMD's q
3:     $\Delta X = R_t(\boldsymbol{\phi}_t(I, \boldsymbol{X}_{t-1}))$      ⇒ apply regressor $\boldsymbol{R}_t$
4:     $\boldsymbol{X}_t = \boldsymbol{X}_t + \Delta\boldsymbol{X}$             ⇒ update shape {Eqn. (4.14)}
5:     update $\Delta\mathbf{p}$ with $\Delta\boldsymbol{X}$        ⇒ {Eqn. (4.16)}
6:     $\boldsymbol{X}_t \approx \boldsymbol{X}_t + \mathbf{J}_t\Delta\mathbf{p}$          ⇒ update shape w.r.t: Eqn. (4.10)
7: end for
```

4.6.2 Experimental Results

Experiments on the Helen dataset [21] show that elBCLM achieves an alignment error of 5.88, at 150 FPS on 68 facial landmarks. Table-4.1 tabulates the fitting accuracy, as well as other state-of-the-art algorithms. We can see that elBCLM outperforms the state-of-the-art algorithms. Fig. 4-7 illustrates fitting results based on elBCLM, LBF [1], and CLM [27], which show elBCLM can locate landmarks more accurately.

Method	**Error (68 landmarks)**
Zhu et. al [53]	8.16*
DRMF [31]	6.70*
RCPR [52]	5.93*
Tadas et. al [27]	6.75
elBCLM	5.88

Table-4.1: Alignment comparision, results from original papers with "*".

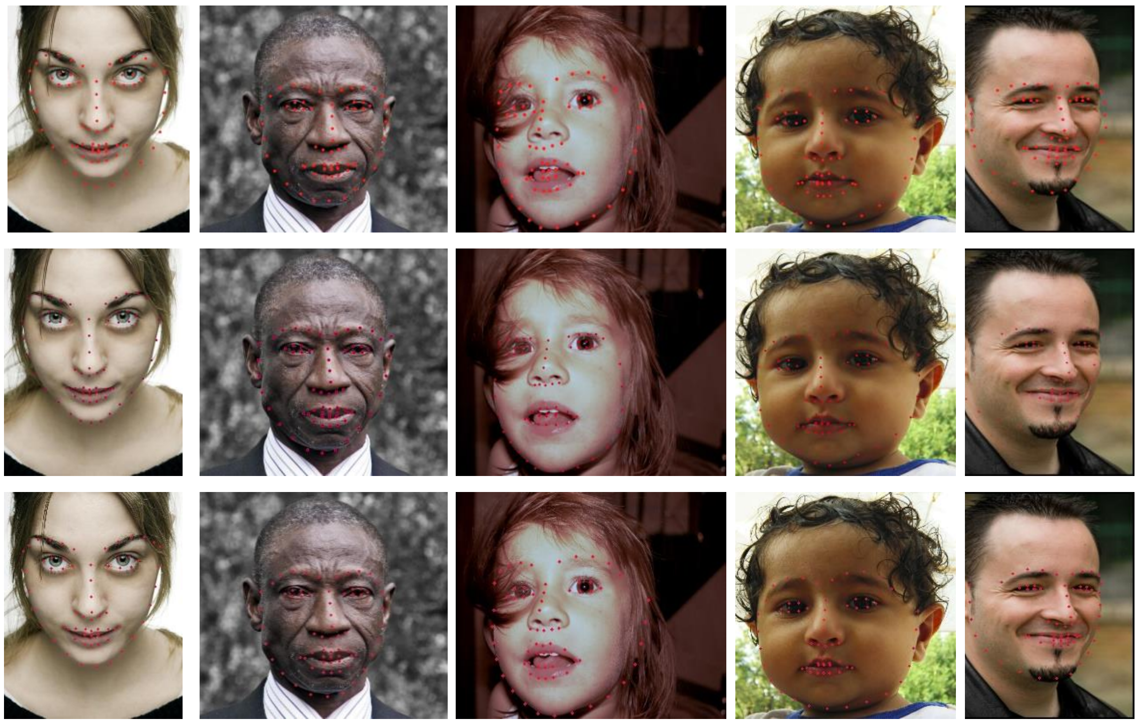


Fig. 4-7: Fitting results comparison (68 points, Helen dataset [21]), Row-1:LBF[1]: Row-2:CLM[27], Row-3: elBCLM

4.7 Conclusions

In this chapter, we propose a more accurate and efficient face alignment algorithm. There are two main contributions in our algorithm. The first one is that we connect two schools of face alignments into a single framework. The second one is that the PDM non-rigid local parameters are used as a guiding, discriminative feature for the cascade alignment framework. To further improve the efficiency, we also propose a feature-switching scheme used in the cascaded framework. Experiment results show that our proposed algorithm outperforms the state-of-the-art algorithms in terms of accuracy and efficiency.

Chapter 5. Joint Maximum Purity Forest with Application to Image Super-Resolution

In this chapter, we propose a novel random-forest scheme, namely Joint Maximum Purity Forest (JMPF), for classification, clustering, and regression tasks. In the JMPF scheme, the original feature space is transformed into a compactly pre-clustered feature space, via a trained rotation matrix. The rotation matrix is obtained through an iterative quantization process, where the input data inclined to different classes are clustered to the respective vertices of the new feature space with maximum purity. In the new feature space, orthogonal hyperplanes, which are employed at the split-nodes of decision trees in random forests, can tackle the clustering problems effectively. We evaluated our proposed method on public benchmark datasets for regression and classification tasks, and experiments showed that JMPF remarkably outperforms other state-of-the-art random-forest-based approaches. Furthermore, we applied JMPF to image super-resolution specifically, because the transformed, compact features are more discriminative to the clustering-regression scheme. Experiment results on several public benchmark datasets also showed that the JMPF-based image super-resolution scheme is consistently superior to recent state-of-the-art image super-resolution algorithms.

5.1 Introduction

Recently, random forest [14, 63] has been employed as an efficient classification or regression tool on a large variety of computer-vision applications, such as object classification [110], recognition [87], face alignment [1, 15, 51], data clustering [17], image super-resolution [8, 19], and so on. This method is attractive on computer-vision problems, not only for its simple implementation, but also for its merits: (1) it can work efficiently on both training and inference stages, (2) it is feasible for it to be sped up with parallel processing technology, (3) it has an inherent property to handle high-dimensional input features, and (4) it works with divide-and-conquer strategy, which has stable performance on classification and regression tasks as an ensemble machine-learning tool. Random forest is a machine-learning method using an ensemble of randomized decision trees, and each tree consists of split-nodes and leaf-nodes, which can be trained recursively. During the training process, at each split-node in a decision tree, a hyperplane is learned to separate data into two groups. Although each decision tree attempts to achieve maximum purity, i.e., maximizes inter-class variance and minimizes intra-class variance, for the two data groups clustered at each split-node independently during training a random forest, there is no guarantee that the original feature space can meet the expectation of global maximum purity for all the clustered groups. As the hyperplanes in a random forest have the orthogonal constraint, as shown in Fig. 5-1(b), which hinders us from achieving the optimal hyperplanes as SVM (support vector machine) [68, 70] does in some original feature space, as shown in Fig. 5-1(a). In this chapter, we aim to solve this orthogonal-constraint limitation. With the fixed orthogonal hyperplanes, we propose to rotate the feature space, this is equivalent to rotating the hyperplanes, in such a way that global maximum purity on the clustered data can be achieved, as illustrated in Fig. 5-2. This strategy can achieve a joint maximum purity for all the split-nodes when training a random forest. Summarily, we propose a feature preprocessing, where the original features are transformed into a new feature space via a constructed rotation matrix, then the performance of constructed random forest can be enhanced in the new feature space. Similar idea can be found in the Rotation Forests [110], but the construction processes on the rotation matrices are different. Moreover, Rotation Forests [110] essentially has restricted itself to classification task, while JMPF keeps the original capability of random forest.

Image super-resolution can be performed based on clustering/classification, according to the recent emerging clustering-regression stream [2, 5, 8], and JMPF scheme can achieve remarkable performance on both the classification and regression tasks. Therefore, JMPF is applied to single-image super-resolution in this chapter. In our algorithm, principal component analysis (PCA) is applied to the features for dimensionality reduction. The projected feature space is then rotated to a compact, pre-clustered feature space via a learned rotation matrix. Finally, for all the split-nodes trained for a random forest, their thresholds are directly set to the inherent zero-center orthogonal hyperplanes in the rotated feature space to meet the maximum-purity criterion. Experiment results show that JMPF can achieve more accurate clustering performance, and applying JMPF to image super-resolution can achieve superior quality, compared to state-of-the-art methods.

Having introduced the main idea of our proposed algorithm, the remainder of this chapter is organized as follows. In Section 2, we will describe our proposed scheme, the JMPF scheme, and present in detail how to compute the rotation matrix via clustering data into the feature-space vertices. Section 3 will evaluate our proposed method and compare its performance with recent state-of-the-art random-forest-based approaches on regression and classification tasks. In Section 4, we will validate the performance of JMPF scheme on single-image super-resolution. Conclusions are given in Section 5.

5.2 Joint Maximum Purity Forest Scheme

5.2.1 Random Forest and Our Insights

In mathematical expression, a random forest is an ensemble of T binary decision trees $\mathcal{T}^t(x): X \rightarrow \mathbb{R}^d$, where $t\ (= 1, 2, \dots, T)$ is the index of the trees, $X \in \mathbb{R}^m$ is the m-dimension feature space, and $\mathbb{R}^d = [0, 1]^d$ represents the space of class probability distributions over the label space $Y = \{1, \dots, d\}$. As shown in Fig. 5-1(b), the vertical dotted line forms a hyperplane, X_1=0, chosen in the first split-node for separating training samples, and the horizontal dotted line is the hyperplane, X_2=0, for the second split-node to cluster all the feature data assigned to this node. This results in separating the three data samples (Red, Green and Blue) into three leaf-nodes.

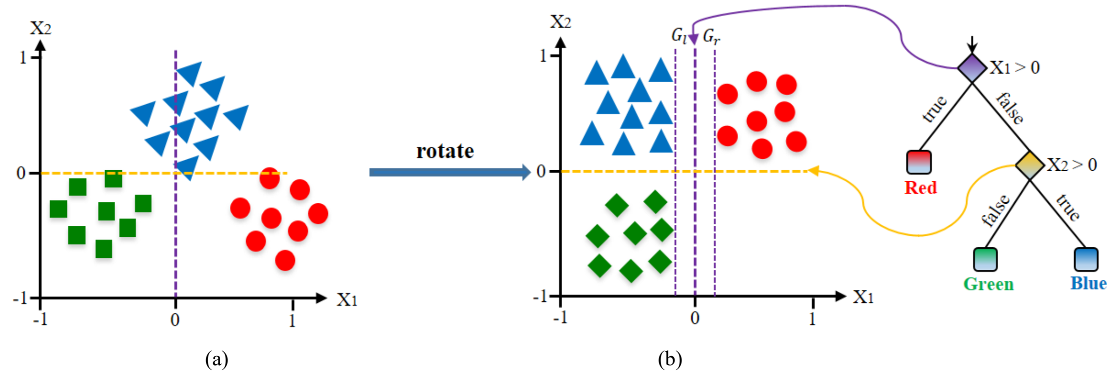


Fig. 5-1: (a) Three classes of samples in a feature space, which are hard to be clustered with orthogonal hyperplanes; and (b) the samples are rotated, and a decision tree of a random forest is used to cluster the data in the new, rotated feature space.

It can be seen from Fig. 5-1(b) that, for each split-node, the optimal hyperplane with more generalization capability is the one which can achieve maximum purity in clustering samples into two groups. For example, the vertical dotted line is the first optimal hyperplane because it clusters all the red training samples into the right node, while all the blue and green samples are clustered into the left node, where the left margin G_l and the right margin G_r are equal.

The training of a whole random forest is to train all of its decision trees, by choosing the candidate features and thresholds for each of the split-nodes, where the feature dimensions and thresholds are determined using a random bagging strategy. In the prediction stage, each decision tree returns a class probability $p_t(y|x)$ for a given query sample $x \in \mathbb{R}^m$, and the final class label y^* is then obtained via averaging, as follows:

$$y^* = \arg\max_y \frac{1}{T}\sum_{t=1}^{T} p_t(y|x). \tag{5.1}$$

The splitting function for a split-node is denoted as $s(v;\Theta)$, where v is a sample and Θ is typically parameterized by two values: (i) a feature dimension $\Theta^i \in \{1,\dots,m\}$, and (ii) a threshold $\Theta^t \in \mathbb{R}$. The splitting function is defined as follows:

$$s(v;\Theta) = \begin{cases} 0, & \text{if } v(\Theta^i) < \Theta^t, \\ 1, & \text{otherwise}, \end{cases} \tag{5.2}$$

where the outcome defines to which child node the sample v is routed, and 0 and 1 are the two labels for the left and right child nodes, respectively. Each node chooses the best splitting function Θ^* out of a randomly sampled set $\{\Theta^t\}$ by minimizing the following function:

$$I = \frac{|L|}{|L|+|R|}H(L) + \frac{|R|}{|L|+|R|}H(R), \tag{5.3}$$

where L and R are the sets of samples that are routed to the left and the right child nodes, and $|S|$ represents the number of samples in the set S. The most important part in Eqn. (5.3) is $H(S)$, which is a criterion to describe the data information in the sample set S. Mathematically, $H(S)$ is the local score for a set of samples (S is either L or R), which normally is calculated using entropy as in Eqn. (5.4), but it can be replaced by variance [1, 18, 51] or the Gini index [14].

$$H(S) = -\sum_{k=1}^{K} \left[p(k|S) * \log\left(p(k|S)\right) \right], \tag{5.4}$$

where K is the number of classes, and $p(k|S)$ is the probability for class k, given the set S. For the regression problem, the differential entropy:

$$H(q) = \int_{y} q(y|x) * \log(q(y|x)) d_y \tag{5.5}$$

over continuous outputs can be employed, where $q(y|x)$ denotes the conditional probability of a target variable given the input sample. Assuming $q(.,.)$ to be a Gaussian distribution and having only a finite set S of samples, the differential entropy can be written in closed form as

$$H_{\text{Gauss}}(S) = \frac{K}{2}(1 - \log(2\pi)) + \frac{1}{2}\log(\det(\Sigma_S)), \tag{5.6}$$

where $\det(\Sigma_S)$ is the determinant of the estimated covariance matrix of the target variables in S. For training each decision tree in a random forest, the goal on each split-node is to maximize the information gain (IG) by reducing the entropy after splitting. IG is defined as follows:

$$\text{IG} = \text{entropy(parent)} - [\text{average entropy(children)}]. \tag{5.7}$$

Since each decision tree is a binary tree and each step is to split a current node (a parent set S) into two children nodes (L and R sets), IG can be described as follows:

$$\arg\max_{\mathcal{H}} IG = \arg\max_{L,R} H(S) - \frac{|L|}{|L|+|R|}H(L) - \frac{|R|}{|L|+|R|}H(R), \tag{5.8}$$

where $\mathcal{H}$ is the optimal hyperplane of the split-node, and Eqn. (5.8) is the target function of each split-node when training each decision tree of a random forest. As we can see from Fig. 5-1(b), all the optimal hyperplanes from split-nodes are achieved independently and locally.

Since each optimal hyperplane is obtained from a subset of feature-dimension candidates with the randomly bagging strategy, there is no guarantee of obtaining a global optimum with respect to all the hyperplanes in all the split-nodes. An intuitive thinking, which was inspired by the data distribution in Fig. 5-1(b), is to achieve a global optimum by jointly considering all the hyperplanes of all the split-nodes, in the form as follows with an intuitively descriptive formula, which can be solved through a greedy approach:

$$\max_{\mathcal{H}_k} IG_{\text{global}} = \arg\max_{\mathcal{H}_k} \prod_{k=1}^{\mathcal{K}} IG_k, \tag{5.9}$$

where $\mathcal{K}$ is the total number of split-nodes that a training sample has routed through a decision tree. As there is no mathematical solution to the problem described in Eqn. (5.9), an alternative way (i.e., an approximate method) to numerically solving Eqn. (5.9) is to jointly maximize the purity of the clustered data groups at each of the split-nodes. This also means that all the data is clustered into the corners (feature-space vertices) of the feature space, as shown in Fig. 5-2.

5.2.2 The Joint Maximum Purity Forest Scheme

By studying the mechanism of a random forest, we can see that the random-forest approach has some critical properties, as do other powerful classifiers, such as SVM (support vector machine) [68, 70] and AdaBoost (short for "Adaptive Boosting") [73]. Both SVM and AdaBoost work as to approximate Bayes decision rule – known to be the optimal classifiers – via minimizing a margin-based global loss function. Each threshold in a decision tree of a random forest works as a hyperplane, and each single decision tree, similar to AdaBoost, attempts to minimize its global loss greedily and recursively on working through from the root-node down to leaf-nodes in the binary tree.

To calculate the threshold for each split-node in each decision tree when training a random forest, we are attempting to determine an orthogonal hyperplane for a three-category classification problem, as shown in Fig. 5-1. Since the hyperplanes for the split-nodes of a decision tree are required to be orthogonal to each other, seeking an optimal orthogonal hyperplane locally cannot guarantee obtaining maximum purity for the whole tree globally. To achieve an optimal classification performance for the whole decision tree, all the split-nodes should be considered globally or simultaneously.

As shown in Fig. 5-2, a number of split-nodes, which have their hyperplanes orthogonal to each other, are required to separate the samples into different nodes. However, if we can transform the samples (zero-centered feature data) to locate them at the respective corners of the feature space, i.e. $\{-1,1\}^m$ for *m*-dimensional features, the feature data can be easily and accurately separated by the orthogonal (either vertical or horizontal) hyperplanes, which contain the space center $\{0\}^m$, as illustrated in Fig. 5-1(b). The insight behind this is that the data is clustered into the feature-space vertices (the corners in a 2-D feature space means that the data points belong to $\{-1,1\}^2$ as the coordinate range is set to $[-1, 1]$).

To tackle the original feature data X, which is not ideally clustered in the vertices or corners of the feature space or close to them, as shown in Fig. 5-1(a), an intuitive idea is to rotate the feature space (this is equivalent to rotating the hyperplanes). This transformation clusters the feature data compactly into m feature-space vertices $\{-1,1\}^m$ with a total of 2^m vertices. Therefore, a possible solution to the problem described in Eqn. (5.10) is to rotate the data features by a rotation matrix $\mathcal{R}^{m\times m}$, as shown in Fig. 5-2, through which the original feature space X is transformed into a more compact clustered feature space, where all the feature data is clustered close to their inclined feature-space vertices B. This solution can be mathematically defined as follows:

$$\min\|B - X\mathcal{R}\|_F^2, \text{ s.t. } B \in \{-1,1\}^{n\times m}, \mathcal{R}^T\mathcal{R} = I \tag{5.10}$$

where $X \in \mathbb{R}^{n\times m}$ contains n samples, each of which is a m-dimensional feature vector arranged in a row, and is zero-centered, i.e. all the feature vectors are demeaned by subtracting the mean vector from each feature vector.

This idea of clustering data into the feature-space vertices can also be found in locality-sensitive hashing (LSH) [61] and image representation [67]. In [61], a simple and efficient alternating minimization scheme was proposed to find a rotation matrix for zero-centered feature data, which minimizes the quantization errors by mapping the feature data to the vertices of a zero-centered binary hypercube. The method is termed as iterative quantization (ITQ), which can work on multi-class spectral clustering and orthogonal Procrustes problem. Yu et al. [55] proposed using a circulant matrix to speed up the computation, because the circulant structure enables the use of Fast Fourier Transformation (FFT). As the computation of the rotation matrix in the training and testing stage is ignorable, we choose a similar scheme to ITQ [61] to determine the rotation matrix $\mathcal{R}$ (we throw away the final quantization matrix B described in Eqn. (5.10), which is used for hashing in [61]), through which the original feature space X can be transformed into a new compact clustered feature space: $\tilde{X} = X\mathcal{R}$, where the data is located at the respective vertices in the new feature space. After this transformation, a random forest with globally joint maximum purity of all the clustered data can be trained, through all the hyperplanes in the split-nodes of each decision tree. Based on this idea, our proposed scheme is called joint maximum purity forest (JMPF).

5.2.3 Rotation Matrix via Clustering Data into Feature-Space Vertices

Assuming that $x \in \mathbb{R}^m$ is one point in the m-dimensional feature space X (zero-centered data), the respective vertices in the zero-centered binary hypercube space can be denoted as $\operatorname{sgn}(x) \in \{-1,1\}^m$, and there is a total of 2^m vertices in the m-dimensional feature space. It is easy to see from Fig. 5-2 that $\operatorname{sgn}(x)$ is the vertex in the feature space, such that it is the closest to x in terms of Euclidean distance. We denote a binary code matrix $B \in \{-1,1\}^{n\times m}$, whose rows $b = \operatorname{sgn}(x) \in B$. For a matrix or a vector, $\operatorname{sgn}(.)$ applies the sign operation to it element-wise.

Our objective is to minimize the error between the feature X and the feature-space vertices B, i.e., $\min\|B - X\|^2$. As we can see in Fig. 5-2, when the feature space is rotated, the feature points will be more concentrated around their nearest vertices, which means that the quantization error will become smaller. Therefore, the minimization problem of $\min\|B - X\|^2$ is equivalent to minimizing the error of the zero-centered data with respect to the Frobenius norm, as in the following formulation:

$$Q(B,\mathcal{R}) = \|B - X\mathcal{R}\|_F^2, \text{ s.t. } B \in \{-1,1\}^{n\times m}, \mathcal{R}^T\mathcal{R} = I. \tag{5.11}$$

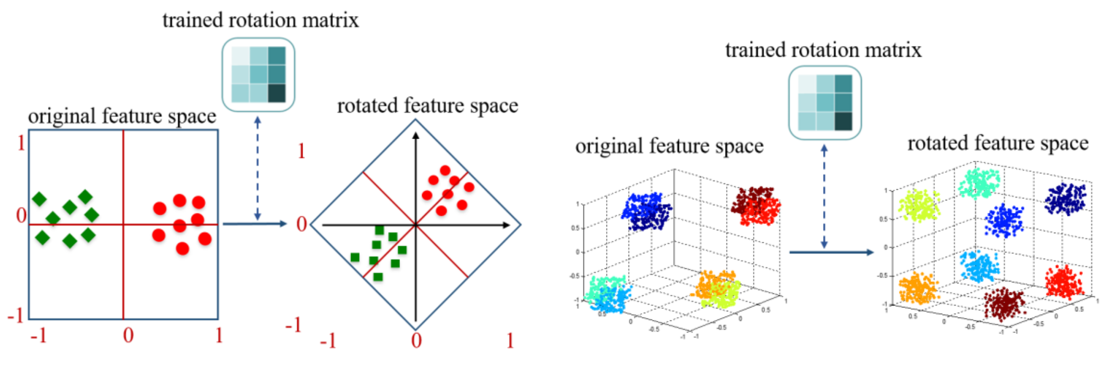


Fig. 5-2: Two toy examples of rotating a feature space into a more compact clustered feature space: (a) 2-dimensional features and (b) 3-dimensional features. The data is clustered into the vertices of a new feature space, by jointly maximizing the purity of all the clustered data.

Therefore, the task of this minimization problem is to determine an optimal rotation matrix $\mathcal{R}$ to satisfy Eqn. (5.11). Since there are two variables in Eqn. (5.11), the expectation–maximization (E-M) algorithm is applied to cluster data into the feature-space vertices, such that a local minimum of the binary code matrix B and the rotation matrix $\mathcal{R}$ are computed simultaneously.

The idea of rotating feature data to minimize the error between the transformed data and the feature-space vertices B can also be found in [67], which showed that the rotation matrix $\mathcal{R}$ can be initialized randomly, and then iterated to converge to the required rotation matrix. Two iteration steps will be performed: in every iteration, each feature vector in the feature space is firstly quantized to the nearest vertex of the binary hypercube, i.e. to a vertex in B, and then the rotation matrix $\mathcal{R}$ is updated to minimize the quantization error by fixing B. These two alternating steps are described in detail below:

(1) *Fix $\mathcal{R}$ and update B*:

$$Q(B,\mathcal{R}) = \|B - X\mathcal{R}\|_F^2$$

$$= \|B\|_F^2 + \|X\|_F^2 - 2tr(B\mathcal{R}^T X^T)$$

$$= n \times m + \|X\|_F^2 - 2tr(B\mathcal{R}^T X^T) \qquad (5.12)$$

Because the zero-centered data matrix X is fixed, minimizing Eqn. (5.12) is equivalent to maximizing the following term:

$$tr(B\mathcal{R}^T X^T) = \sum_{i=1}^{n}\sum_{j=1}^{m} B_{ij}\tilde{X}_{ij} \qquad (5.13)$$

where $\tilde{X}_{ij}$ is an element of $\tilde{X} = X\mathcal{R}$. To maximize Eqn. (5.13) with respect to B, $B_{ij} = 1$ whenever $\tilde{X}_{ij} \geq 0$ and $B_{ij} = -1$ otherwise, i.e. $B = \text{sgn}(X\mathcal{R}) \in \{-1,1\}^m$.

(2) *Fix B and update $\mathcal{R}$*:

The problem of fixing B to obtain a rotation matrix based on the objective function Eqn. (5.11) is relative to the classic orthogonal Procrustes problem [66, 94, 96], in which a rotation matrix is determined to align one point set with another.

In our algorithm, these two point sets are the zero-centered data set X and the quantized matrix B. Therefore, a closed-form solution for $\mathcal{R}$ is available, by applying SVD [148] on the $m \times m$ matrix XB^T to obtain $U\Omega V^T$ (Ω is a diagonal matrix), then set $\mathcal{R} = UV^T$ to update $\mathcal{R}$.

II.4 Proof of the Orthogonal Procrustes Problem:

For completeness, we prove the orthogonal Procrustes problem, for which the solution can be found in [66, 94, 96]. The orthogonal Procrustes problem is a matrix approximation problem. In its classical form, one is given two matrices B and X and asked to find a rotation matrix $\mathcal{R}$ (subject to $\mathcal{R}^T\mathcal{R} = I$) which most closely maps $X\mathcal{R}$ to B as formulated in following,

$$\text{Problem definition:} \quad \min_{\mathcal{R}} \|B - X\mathcal{R}\|_F^2 \quad \text{s.t.:} \quad \mathcal{R}^T\mathcal{R} = I\,. \qquad (5.14)$$

$$\text{Proof:} \quad \|B - X\mathcal{R}\|_F^2 \qquad (5.15)$$

$$= tr(B - X\mathcal{R})(B^T - \mathcal{R}^T X^T)$$
$$= tr(BB^T) - 2tr(BX^T\mathcal{R}^T) + tr(\mathcal{R}XX^T\mathcal{R}^T)$$

Thus, $\min_{\mathcal{R}} \|B - \mathcal{R}X\|_F^2$ equals to maximizing:

$$\begin{aligned}
& tr(BX^T\mathcal{R}^T) && (5.16)\\
&= tr(U\Omega V^T\mathcal{R}^T) && \left(\text{SVD on } BX^T: [U, \Omega, V] = \text{svd}(BX^T)\right)\\
&= tr(\Omega V^T\mathcal{R}^T U) && (denote: Z = V^T\mathcal{R}^T U)\\
&= tr(\Omega Z)\\
&= tr \textstyle\sum_i \mathrm{Z}_{i,i}\Omega_{i,i}\\
&\le \textstyle\sum_i \Omega_{i,i}
\end{aligned}$$

The last inequality holds because Z is also an orthonormal matrix, and $\sum_j Z_{i,j}^2 = 1, Z_{i,i} \le 1$. The objective function can be maximized if Z $= I$, i.e.

$$\mathcal{R} = UV^T \qquad \blacksquare$$

5.3 Joint Maximum Purity Forest for Regression and Classification

5.3.1 The Workflow of Joint Maximum Purity Forest

Most random-forest-based models [1, 18, 83, 84] share the similar workflow, as shown in Fig. 5-3, in which the main task on training a tree in a random forest is to decide thresholds in the split-nodes and learn the regressors or classes in the leaf-nodes. Rigid regression or linear regression is often employed in the leaf-nodes for the prediction task, because rigid regression has a closed-form solution, while linear regression is an efficient optimization tool, and the *LibLinear* package [56] can be used to fine-tune its configurations.

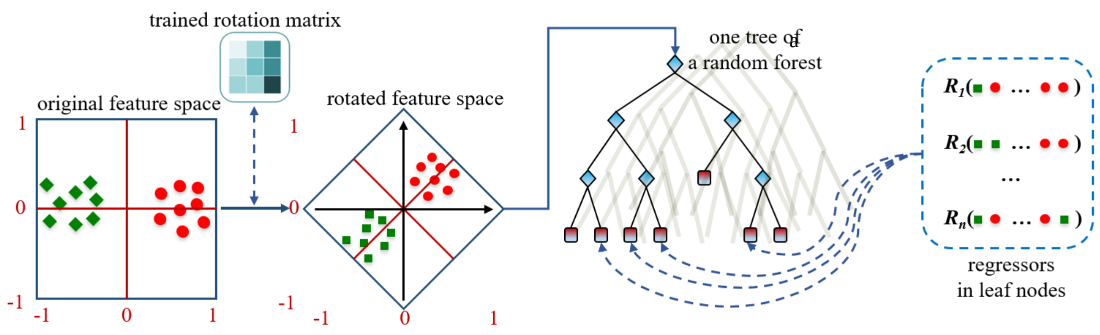


Fig. 5-3: An overview of the workflow of the JMPF-based random forest.

Compared to conventional random forests, our JMPF scheme has one more step, as shown in the left of Fig. 5-3, the rotation matrix. The JMPF scheme transforms the original feature space by rotating it into a more compact, pre-clustered feature space, using a trained rotation matrix learned through clustering feature vectors iteratively into the vertices of a new feature space. The whole workflow of our proposed algorithm, the JMPF scheme, is outlined in Fig. 5-3.

5.3.2 The Inherent Zero-center Hyperplanes as Thresholds for Split-nodes

In training a random forest, the two main operations for training (splitting) each split-node are to choose splitting feature(s), and to determine the threshold, using a random bagging strategy, which can avoid over-fitting in training classifiers. In the rotated compact pre-clustered feature space, the inherent zero-center hyperplanes are inherently the optimal thresholds (to meet the max-purity criterion on two clustered data groups) after training the rotation matrix. Therefore, these inherent zero-center hyperplanes can directly be set as the thresholds to achieve optimal classification performance on training a random forest. Compared to conventional random forests, our proposed JMPF only needs to choose splitting feature(s) to split data at split-nodes. These inherent zero-center hyperplanes can speed up the training process for a random forest, and experimental results in the next subsection will validate their performance.

5.3.3 Experimental Results on JMPF Regression and Classification

To evaluate the performances of the proposed JMPF, we test it with 15 standard machine-learning tasks, 7 for classification and 8 for regression. The datasets used in the experiments are summarized in Table-5.1. We use standard performance evaluation metrics: error rate for classification and root mean squared error (RMSE) for regression, unless otherwise specified.

Dataset	#Train	#Test	#Feature	#Classes or TargetDim
(c)char74k	66707	7400	64	62
(c)gas sensor	11128	2782	128	6
(c)isolet	6238	1558	617	26
(c)letterorig	16000	4000	16	26
(c)pendigits	7494	3498	16	10
(c)sensorless	46800	11700	48	11
(c)usps	7291	2007	256	10
(r)delta ailerons	7129*3/4	7129/4	5	1
(r)delta elevators	5720	3807	6	1
(r)elevators	8752	7847	18	1
(r)kin8nm	8192*3/4	8192/4	8	1
(r)price	159*3/4	159/4	15	1
(r)pyrim	74*3/4	74/4	27	1
(r)stock	950*3/4	950/4	10	1
(r)WiscoinBreastCancer	194*3/4	194/4	32	1

Table-5.1: The properties of the standard machine-learning datasets used for classification and regression. The top 7 are used for classification (c) and the bottom 8 for regression (r). (3/4 means 75% training and 25% testing)

We firstly evaluate the proposed approach on two real applications, one for classification (Table-5.2) and one for regression (Table-5.3). Our proposed JMPF is compared with the original random forest before refinement (denoted as RF), and two state-of-the-art variants: alternating decision forests (ADF) [83] and alternating regression forests (ARF) [84], for classification and regression, respectively.

Furthermore, we compare with JMPF+ADF/ARF, for demonstrating that our algorithm can be combined with other methods. We follow the experiment settings in [83, 84]. We set the maximum tree depth at 15, and the minimum sample number in a splitting node is set at 5. The experiments were repeated five times, and the average error and standard deviation were measured.

dataset	$\#\mathcal{H}$	RF	ADF	JMPF	JMPF+ADF	λ
	1	2.261±0.021	2.173±0.014	**2.147**±0.021 **(05%)**	2.114±0.016 **(07%)**	
char74k ch	3	2.449±0.029	2.236±0.015	**2.206**±0.027 **(10%)**	2.143±0.024 **(12%)**	10^{-1}
	5	2.452±0.016	2.232±0.021	**2.209**±0.019 **(10%)**	2.138±0.017 **(13%)**	
	1	5.656±0.534	5.238±0.539	**4.211**±0.252 **(26%)**	3.958±0.508 **(30%)**	
gas sensor	3	6.264±0.042	5.952±0.323	**4.622**±0.299 **(26%)**	4.416±0.370 **(30%)**	10^{-3}
	5	6.470±0.332	5.751±0.792	**4.775**±0.459 **(26%)**	4.159±0.324 **(36%)**	
	1	6.932±0.281	6.208±0.338	**6.153**±0.381 **(11%)**	5.868±0.239 **(15%)**	
isolet	3	6.501±0.199	6.308±0.330	**6.272**±0.332 **(04%)**	5.932±0.177 **(09%)**	10^{-2}
	5	7.005±0.362	6.528±0.261	**6.381**±0.254 **(09%)**	5.969±0.205 **(15%)**	
	1	6.371±0.099	4.418±0.082	**4.114**±0.087 **(35%)**	3.535±0.111 **(45%)**	
letterorig	3	6.889±0.199	5.196±0.127	**4.864**±0.267 **(29%)**	4.146±0.192 **(40%)**	10^{-2}
	5	6.739±0.263	5.082±0.097	**4.625**±0.257 **(31%)**	4.032±0.131 **(40%)**	
	1	3.528±0.124	3.234±0.106	**2.912**±0.069 **(17%)**	2.850±0.136 **(19%)**	
pendigits	3	3.418±0.171	3.377±0.164	**2.969**±0.120 **(13%)**	2.915±0.100 **(15%)**	10^{-2}
	5	3.499±0.184	3.283±0.184	**3.054**±0.081 **(13%)**	3.002±0.086 **(14%)**	
	1	1.824±0.018	0.972±0.028	**0.324**±0.005 **(82%)**	0.253±0.009 **(86%)**	
sensorless	3	1.026±0.158	0.391±0.007	**0.293**±0.004 **(71%)**	0.281±0.003 **(73%)**	10^{-1}
	5	0.903±0.150	0.512±0.223	**0.268**±0.054 **(70%)**	0.244±0.029 **(73%)**	
	1	6.128±0.181	6.149±0.208	**6.085**±0.216 **(01%)**	5.964±0.206 **(03%)**	
usps	3	6.527±0.203	6.520±0.188	**6.285**±0.101 **(04%)**	6.206±0.245 **(05%)**	10^{-2}
	5	6.548±0.225	6.441±0.195	**6.391**±0.063 **(02%)**	6.213±0.112 **(05%)**	

Table-5.2: Comparison of classification performances on seven datasets, which can be found at UCI machine-learning repository: https://archive.ics.uci.edu/ml/datasets.html. RF: standard random forest, ADF: alternating decision forests [83], JMPF: proposed algorithm, JMPF+ADF: our proposed algorithm embedded into ADF. $\#\mathcal{H}$ is the number of randomly chosen hyperplane(s) on training a random forest. λ is the error scale. The percentages in brackets for JMPF and JMPF+ADF are the reduction rates in RMSE (root mean squared error) compared with the RF algorithm.

The results are presented in Table-5.2 and Table-5.3, for the classification and regression tasks, respectively. In terms of accuracy, our proposed JMPF significantly outperforms the standard random forest on all classification and regression tasks.

Compared to RF, JMPF achieves an average of 23.57% improvement on the classification tasks, and an average of 23.13% improvement on the regression tasks. Our method also consistently outperforms the state-of-the-art variants: ADF/ARF. Moreover, the performance of our JMPF algorithm can be further improved by integrating with ADF and ARF, denoted as JMPF + ADF/ARF. As shown in Table-5.2 and Table-5.3, JMPF+ADF achieves an average 27.86% improvement on the classification tasks, while JMPF+ARF achieves an average 26.88% improvement on the regression tasks. These results on diverse tasks clearly demonstrate the effectiveness of our proposed approach.

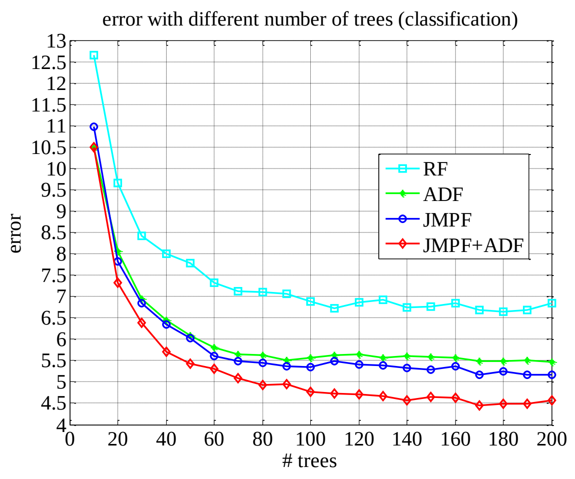
dataset	RF	ARF	JMPF	JMPF+ARF	λ
delta ailerons	2.970±0.001	2.967±0.006	**1.952**±0.003 **(34%)**	1.946±0.002 **(34%)**	10^{-4}
delta elevators	2.360±0.002	2.338±0.008	**1.635**±0.001 **(30%)**	1.610±0.006 **(32%)**	10^{-3}
elevators	0.638±0.001	0.635±0.001	**0.619**±0.001 **(03%)**	0.606±0.001 **(05%)**	10^{-2}
kin8nm	2.622±0.002	2.545±0.003	**1.962**±0.003 **(25%)**	1.667±0.005 **(36%)**	10^{-1}
price	7.281±0.755	6.663±0.794	**5.460**±0.627 **(25%)**	5.234±0.666 **(28%)**	10^{1}
pyrim	1.440±0.008	1.042±0.347	**1.031**±0.017 **(28%)**	0.631±0.018 **(56%)**	10^{-1}
stock	2.878±0.022	2.823±0.038	**2.744**±0.019 **(05%)**	2.678±0.021 **(07%)**	10^{0}
Wiscoin breast cancer	3.669±0.041	3.130±0.044	**3.081**±0.008 **(16%)**	3.036±0.023 **(17%)**	10^{1}

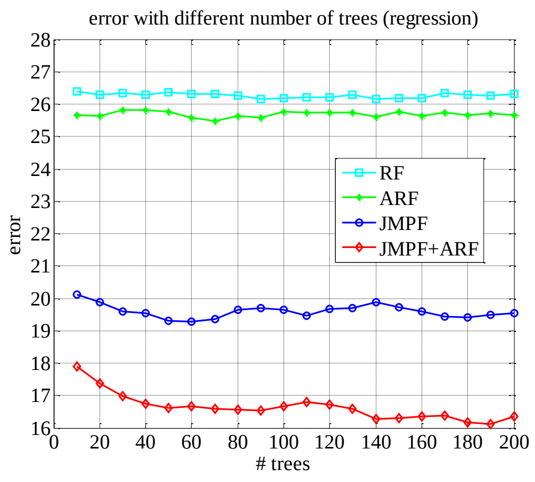
Table-5.3: Comparison of regression performances on eight datasets, which can be found at http://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html. RF: standard random forest, ARF: alternating regression forests [84], JMPF: proposed algorithm, JMPF+ARF: our proposed algorithm embedded into ARF. λ is the error scale. The number of randomly chosen hyperplanes $\#\mathcal{H}$ is 3. The percentages in brackets for JMPF and JMPF+ARF are the reduction rates in RMSE compared with the RF algorithm.

5.3.4 Discussions on Experimental Results

The computational complexity of JMPF is similar to that of the standard random forest. As illustrated in the workflow of JMPF in Fig. 3, only one additional step, which computes the rotation matrix, is required, when compared to the standard random forest. For a small dataset (e.g., feature dimension size less than 500 and data size less than

10,000), the computation required to compute the rotation matrix for clustering data into the feature-space vertices is acceptable in the training stage (around 10 seconds by using MatLab) and negligible in the testing stage. When the dimension size becomes larger, PCA dimensionality reduction can be employed. If the size of the dataset increases, such that using PCA still involves heavy computation, bagging can be used to achieve comparable accuracy and the whole extra computation will be insignificant.





(a) (b)

Fig. 5-4: Performance with different numbers of trees for (a) classification and (b) regression (dataset for classification is *letterorig* and dataset for regression is *kin8nm*, error scale: 10^{-2}, the number hyperplane(s) $\#\mathcal{H}$ on training random forest is 3).

To study the stability of JMPF, we choose the *letterorig* dataset for classification and the *kin8nm* dataset for regression, and the respective results are shown in Fig. 5-4(a) and Fig. 5-4(b), respectively. In the experiments, the number of trees, i.e., the number of weak classifiers in the random forest, varies from 10 to 200, and we have three observations. Firstly, as shown in Fig. 5-4, when the number of trees increases, the performance of all the algorithms improves. For classification, as shown in Fig. 5-4(a), when the number of trees is larger than 100, the errors are converged to become steady. On the contrary, for the regression task as shown in Fig. 5-4(b), the errors are almost stable, ranged from 10 to 200. Secondly, the results show that JMPF consistently outperforms ADF and RF, irrespective of the number of trees used. Finally, Fig. 5-4 clearly shows that JMPF can integrate with ADF or ARF to further improve its performance.

5.4 Image Super-Resolution via Joint Maximum Purity Forest

5.4.1 JMPF-based Image Super-Resolution

The recent emerging stream [65, 91] on single-image SR is to formulate the problem as a clustering-regression problem, which can be solved with machine-learning tools. These approaches are learning-based methods, which attempt to reconstruct an HR image from patches with the help of an external database. These methods first decompose an image into patches, then classify them into clusters. Regressors are then trained for each of the clusters, which generate mappings from an input LR patch's feature to its corresponding HR patch (see Fig. 5-5). In the testing stage, an LR query image follows the same procedures to cut into patches and to extract features, which are then assigned to their corresponding clusters using the *k*-NN algorithm [8, 19] or random forest [62, 65, 67]. The respective HR patches are constructed through regressors learned for the clusters (see Fig. 5-6). This kind of clustering-regression algorithms, based on random forest [62, 65, 67], has achieved state-of-the-art performance in single image super-resolution, both in terms of accuracy and efficiency, because of the use of ensemble learning and sublinear search. As JMPF achieves promising results on both classification and regression tasks, it can be employed for image super-resolution for better performances.

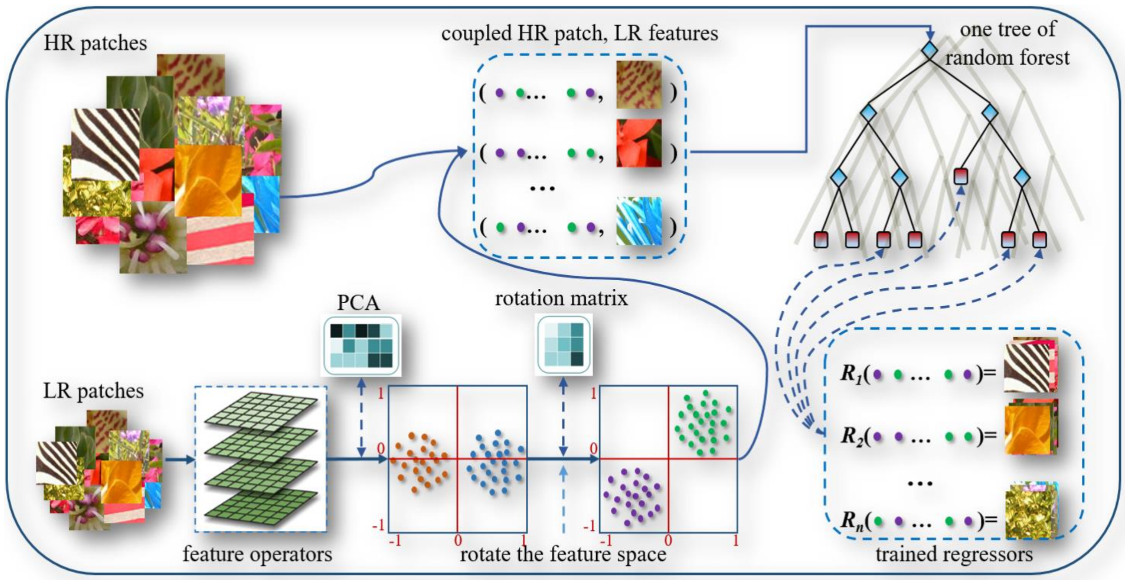


Fig. 5-5: An overview of the training process of the JMPF-based image super-resolution.

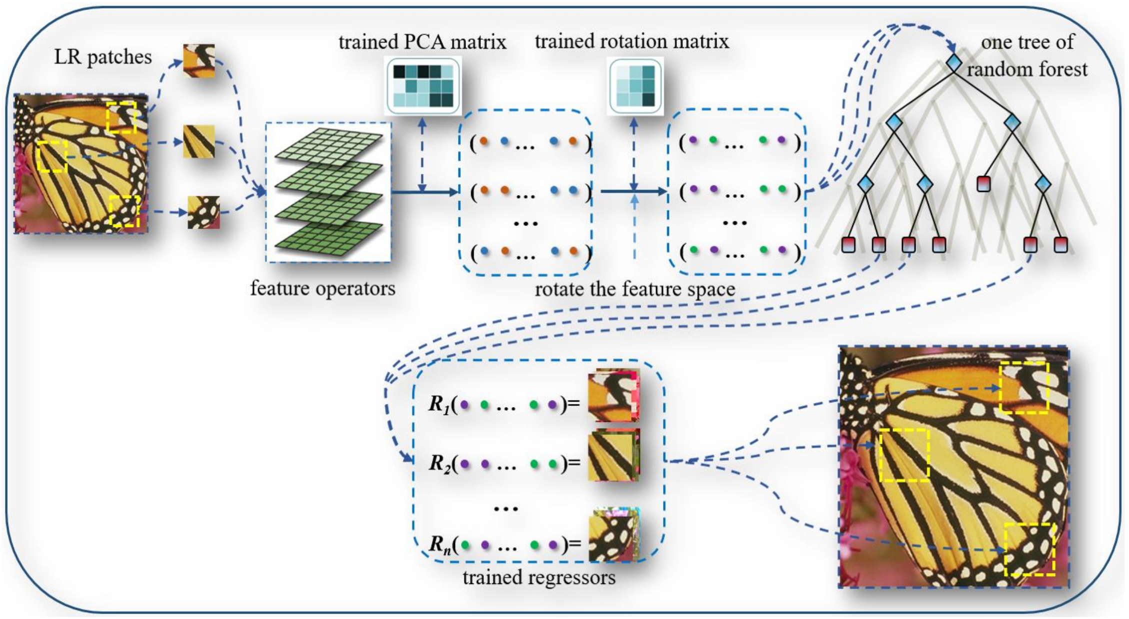


Fig. 5-6: An overview of the testing process of the JMPF-based image super-resolution

An overview of the training and testing processes of the proposed JMPF-based image SR method is illustrated in Fig. 5-5 and Fig. 5-6, respectively. In our method, the first and second-order gradients are extracted as features from each patch, followed by PCA for dimensionality reduction. These features are then rotated into a more compact, pre-clustered feature space. Finally, all the thresholds are directly set to the inherent zero-center hyperplanes when training the random forest, and similar to other algorithms, the regressors at the leaf-nodes are computed using the rigid regression algorithms. This approach is named as JMPF-based image super-resolution method.

5.4.2 The Working Processes of JMPF-based Image Super-Resolution

JMPF has been shown to achieve a better performance for clustering and classification than other random forest methods. Since image super-resolution can be considered as a clustering/classification problem, using JMPF is likely to result in better performance. This is mainly due to the features transformed to the vertices in the new feature space, so the features become more discriminative. The image super-resolution training and testing processes of our proposed JMPF-based method are described in Algorithm 5.1 and Algorithm 5.2, respectively.

Algorithm 5.1: JMPF-based Image Super-Resolution Training Process:

Input: $\{x_i^l, x_i^h\}_{i=1}^N$: training LR-HR patch pairs, N is the number of training samples.

Output: the random forest and ridge regression projection matrices: $\wp = (P_1, \dots, P_T)$, in leaf- nodes, where T is the number of regressors; the PCA projection matrix $\mathcal{M}$ and the rotation matrix $\mathcal{R}$.

```
1: Discriminative features calculated from patch images based on first and second-
order (horizontal and vertical) gradients;                    ⇒ {Eqn. (2.15)}
2: Apply PCA on features to compute the PCA projection matrix $\mathcal{M}$;
3: Train a JMPF-based random forest by clustering PCA projected feature data into
feature-space vertices, which can rotate the feature space into a compact pre-
clustered feature space, as well obtain the rotation matrix $\mathcal{R}$;      ⇒ {Eqn. (5.11)}
4: Train ridge regression projection matrices: $\wp = (P_1, \dots, P_T)$, from LR-HR patch
pairs in all the leaf-nodes.                                   ⇒ {Eqn. (2.21)}
```

Algorithm 5.2: JMPF-based Image Super-Resolution Testing Stage:

```
Input: testing LR image $I^l$, the trained JMPF-based random forest and ridge regression projection matrices: $\wp = (P_1, \dots, P_T)$ in leaf-nodes; the trained PCA projection matrix $\mathcal{M}$ and the trained rotation matrix $\mathcal{R}$.
Output: super-resolved image $I^h$.
1: Extract discriminative features for all the patches of image $I^l$;    $\Rightarrow$ {Eqn. (2.15)}
2: Do feature dimension reduction via the PCA projection matrix $\mathcal{M}$;
3: Rotate feature space into a compact pre-clustered feature space via the rotation matrix $\mathcal{R}$;
4: For LR patches from image $I^l$, based on their features, searching their corresponding regressors from leaf-nodes in the trained random-forest;
5: Produce $I^h$ through all the image patches from image $I^l$ by ridge regression with the trained projection matrices: $\wp = (P_1, \dots, P_T)$.    $\Rightarrow$ {Eqn. (2.20)}
```

5.4.3 Experimental Results on JMPF-based Image Super-Resolution

In this section, we evaluate our image SR algorithm on some standard image super-resolution datasets, including Set 5, Set14, and B100 [80], and compare it with a number of classical or state-of-the-art methods. These include conventional bicubic interpolation, sparse representation SR (Zeyde) [64], anchored neighborhood regression (ANR) [62], A+ [65], standard random forest (RF) [18], and alternating regression forests (ARF) [18]. We set the same parameters for all the random-forest-based algorithms: the number of trees in the random forest is 10, and the maximum depth of each tree is 15.

Dataset	*scale*	bicubic	Zeyde	ANR	A+	RF	ARF	JMPF$^-$	JMPF	**JMPF$^+$**
Set5	×2	33.66	35.78	35.83	36.55	36.52	**36.65**	36.53	36.59	***36.70***
	×3	30.39	31.92	31.93	**32.59**	32.44	32.53	32.51	**32.59**	***32.67***
	×4	28.42	29.74	29.74	**30.28**	30.10	30.17	30.14	30.17	***30.24***
Set14	×2	30.23	31.81	31.80	32.28	32.26	**32.33**	32.27	32.32	***32.42***
	× 3	27.54	28.68	28.66	**29.13**	29.04	29.10	29.12	**29.13**	***29.24***
	×4	26.00	26.88	26.85	**27.33**	27.22	27.28	27.29	27.30	***27.37***
B100	×2	29.32	30.40	30.44	30.78	31.13	31.21	31.16	**31.23**	***31.31***
	×3	27.15	27.87	27.89	28.18	28.21	28.26	28.26	**28.30**	***28.37***
	×4	25.92	26.51	26.51	26.77	26.74	26.77	26.78	**26.81**	***26.87***

Table-4: Results of the proposed method, compared with state-of-the-art methods on 3 datasets, in terms of PSNR (dB), with three different magnification factors (×2, ×3, ×4).

Experiment results are tabulated in Tables-5.4 and Tables-5.5, where JMPF is our proposed JMPF-based image super-resolution method, and JMPF⁻ is a trimmed version, such that the thresholds for the split-nodes are not the inherent zero-center hyperplanes, but set by the standard random-forest bagging strategy. We use the same training images (91 images) for all the algorithms as previous works [62, 64, 65, 18] do. However, for JMPF^{+}, 100 more images from the General-100 dataset [82] are used, so as to check whether or not more training samples can further improve our proposed algorithm.

Set5(×2)	bicubic	Zeyde	ANR	A+	RF	ARF	JMPF^{-}	JMPF	**JMPF^{+}**
baby	37.05	38.22	38.42	**38.52**	38.47	38.48	38.40	38.45	***38.45***
bird	36.82	39.91	40.03	41.06	40.98	**41.15**	40.82	40.99	***41.11***
butterfly	27.43	30.64	30.54	32.02	32.27	**32.66**	32.58	32.50	***32.79***
head	34.85	35.62	35.72	35.82	35.69	35.73	35.68	**35.73**	***35.78***
woman	32.14	34.53	34.53	35.31	35.19	35.24	35.15	**35.28**	***35.38***
average	33.66	35.78	35.85	36.55	36.52	**36.65**	36.53	36.59	***36.70***

Set5(×3)	bicubic	Zeyde	ANR	A+	RF	ARF	JMPF^{-}	JMPF	**JMPF^{+}**
baby	33.91	35.13	35.13	35.23	**35.25**	35.15	35.11	35.16	***35.14***
bird	32.58	34.62	34.63	**35.53**	35.23	35.31	35.25	35.46	***35.49***
butterfly	24.04	25.93	25.92	27.13	27.00	27.39	27.46	**27.48**	***27.73***
head	32.88	33.61	33.64	33.82	33.73	33.73	33.72	**33.79**	***33.76***
woman	28.56	30.32	30.31	31.24	30.98	**31.08**	31.03	31.06	***31.24***
average	30.39	31.92	31.93	**32.59**	32.44	32.53	32.51	**32.59**	***32.67***

Set5(×4)	bicubic	Zeyde	ANR	A+	RF	ARF	JMPF^{-}	JMPF	**JMPF^{+}**
baby	31.78	33.13	33.07	**33.3**	33.26	33.16	33.09	33.12	***33.12***
bird	30.18	31.75	31.82	**32.5**	32.21	32.26	32.27	32.33	***32.47***
butterfly	22.10	23.67	23.58	24.4	24.32	**24.56**	24.55	24.44	***24.63***
head	31.59	32.23	32.34	**32.5**	32.35	32.37	32.35	32.45	***32.47***
woman	26.46	27.94	27.88	**28.6**	28.38	28.48	28.44	28.50	***28.53***
average	28.42	29.74	29.74	**30.28**	30.10	30.17	30.14	30.17	***30.24***

Table-5.5: Detailed results of the proposed method, compared with state-of-the-art methods on the dataset Set5, in terms of PSNR (dB) using three different magnification factors (×2, ×3, ×4).

Table-5.4 tabulates the performances, in terms of the average peak signal to noise ratio (PSNR) scores, of our proposed algorithm and other image SR methods, on the 3 datasets with different magnification factors. For the Set5 and Set14 datasets, with different magnification factors, our proposed JMPF-based algorithm can achieve a comparable performance to other recent state-of-the-art methods, such as A+ and ARF.

As those random-forest-based algorithms may not be stable on small datasets, when evaluation works on extensive datasets, such as B100, our proposed algorithm JMPF can stably outperform A+ and ARF for all magnification factors (×2, ×3, ×4). Moreover, the objective quality metrics on PSNR also show that the JMPF algorithm can achieve a better performance when more samples are used for training, as shown from JMPF$^+$ in Table-5.4. Table-5.5 provides more details of the performances in datasets Set5.

To compare the visual quality of our proposed JMPF-based SR algorithm to other methods, Fig. 5-7, shows the reconstructed HR images using different methods. Some regions in the reconstructed images are also enlarged, so as to show the details in the images. In general, our proposed method can produce better quality images, particularly in areas with rich texture, which verifies the feature discrimination of the proposed JMPF scheme.

5.5 Conclusions

In this section, we have proposed a novel random-forest scheme, namely the Joint Maximum Purity Forest (JMPF) scheme, which rotates the feature space into a compact, clustered feature space, by jointly maximizing the purity of all the feature-space vertices. In the new pre-clustered feature space, orthogonal hyperplanes can be effectively used in the split-nodes of a decision tree, which can improve the performance of the trained random forest. Compared to the standard random forests and the recent state-of-the-art variants, such as alternating decision forests (ADF) [83] and alternating regression forests (ARF) [84], our proposed random-forest method inherits the merits of random forests (fast training and testing, multi-class capability, etc.), and also yields promising results on both classification and regression tasks. Experiments have shown that our method achieves an average improvement of about 20% for classification and regression on publicly benchmarked datasets. Furthermore, our proposed scheme can integrate with other methods, such as ADF and ARF, to further improve the performance. The source code of our algorithm is available to download at: https://github.com/HarleyHK/JMPF.

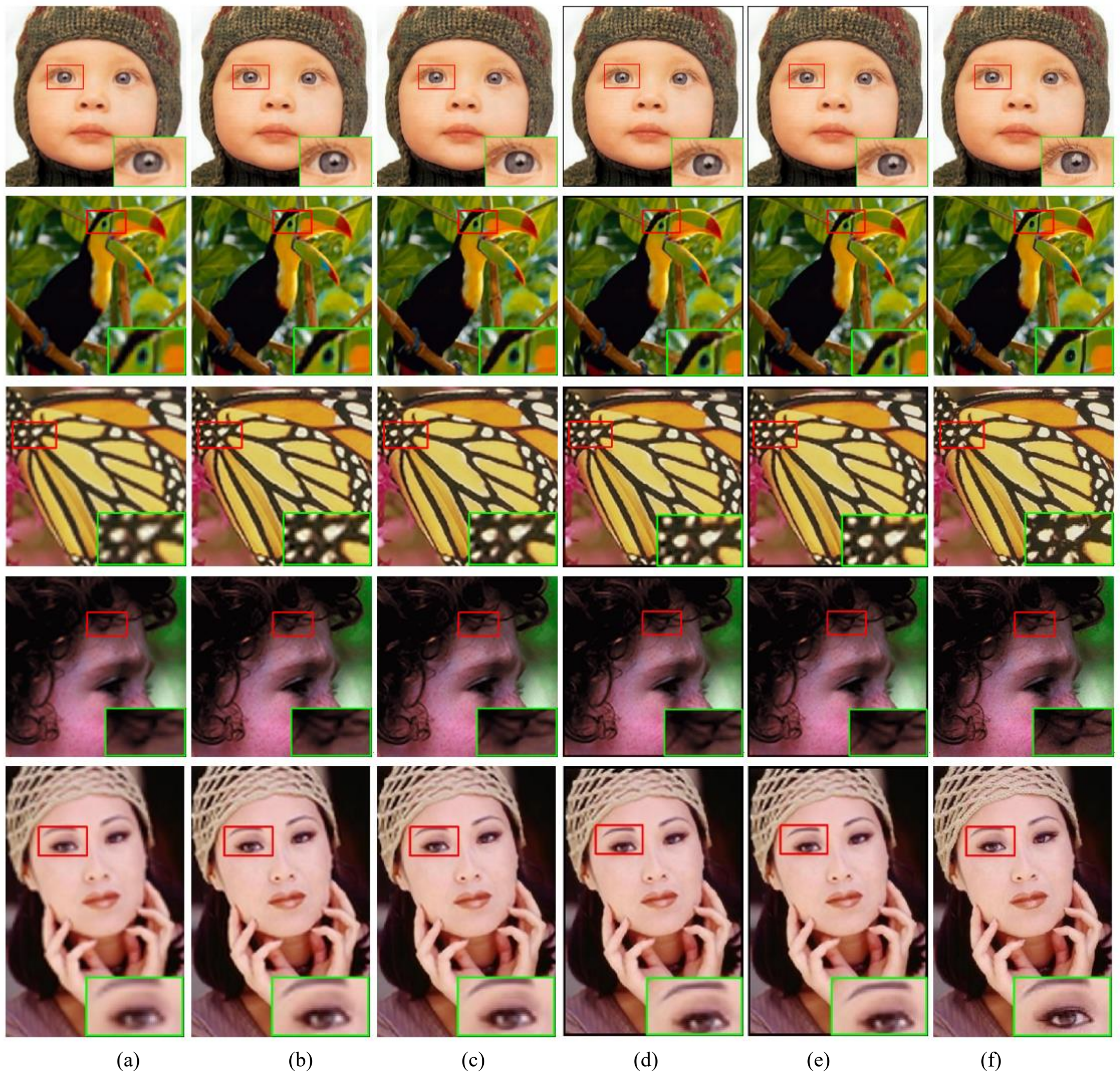


Fig. 5-7: Super-resolved (×3) images from Set5: (a) bicubic, (b) ANR[62], (c) A+[65], (d) ARF[18], (e) proposed algorithm JMPF, and (f) ground truth. The results show that our JMPF-based algorithm can produce more details.

We have also applied JMPF to single-image super-resolution specifically. We tackle image super-resolution as a clustering-regression problem, and focus on the clustering stage, which happens at the split-nodes of each decision tree. By employing the JMPF strategy, we rotate the feature space into a pre-clustered feature space, which can cluster samples into different sub-spaces more compactly in an unsupervised problem. The compact pre-clustered feature space can provide the optimal thresholds for split-nodes in decision trees, which are the zero-centered orthogonal hyperplanes. Our experiment results on intensive image benchmark datasets, such as B100, show that the proposed JMPF-based image super-resolution approach can consistently outperform recent state-of-the-art algorithms, in terms of PSNR and visual quality. Our method also inherits the advantages of random forests, which have fast speed on both the training and inference processes.

Chapter 6. Image Super-Resolution via Feature-Augmented Random Forest

Recent random-forest (RF)-based image super-resolution approaches inherit some properties from dictionary-learning-based algorithms, but the effectiveness of the features working in RF is overlooked in the literature. In this section, we present a novel feature-augmented random forest (FARF) method for image super-resolution, where the conventional gradient-based features are proposed to augment the features used in RF, and different feature recipes are formulated on different processing stages in an RF. The advantages of our method are that, firstly, the dictionary-learning-based features are enhanced by adding gradient magnitudes, based on the observation that the non-linear gradient magnitudes are highly discriminative. Secondly, generalized locality-sensitive hashing (LSH) is used to replace principal component analysis (PCA) for feature dimensionality reduction in constructing the trees, but the original high-dimensional features are employed, instead of the compressed LSH features, for the leaf-nodes' regressors. With the use of the original higher dimensional features, the regressors can achieve better learning performances. Finally, we present a generalized weighted ridge regression (GWRR) model for the leaf-nodes' regressors. Experiment results on several public benchmark datasets show that our FARF method can achieve an average gain of about 0.3 dB, compared to traditional RF-based methods. Furthermore, a fine-tuned FARF model can compare to, or (in many cases) outperform, some recent state-of-the-art deep-learning-based algorithms.

6.1 Introduction

In the past few years, random forest (RF) [63, 14], a machine-learning tool, working via an ensemble of multiple decision trees, has been employed for efficient classification and regression problems, and applied to a large variety of computer-vision applications, such as object recognition [87], face alignment [15, 51, 1], data clustering [17], single image super-resolution (SISR) [18, 19], and so on.

In the past few years, random forest (RF) [63, 14], a machine-learning tool, working via an ensemble of decision trees, has been employed for efficient classification and regression problems, and applied to a large variety of computer-vision applications, such as object recognition [87], face alignment [15, 51, 1], data clustering [17], single image super-resolution (SISR) [18, 19], and so on. The RF-based SISR approach can be considered as a clustering/classification-based method, as shown in Fig. 6-1. However, the clustering and regression problems in RF require different discriminative features, which have not been systematically studied in the literature.

Feature engineering is a research hotspot on the image-restoration problems. Pioneer work in [75] used a simple high-pass filter, which is simply subtracting the output of a low-pass filter from the input image. Meanwhile, most algorithms [62, 97, 64, 65, 8] follow the approach in [88], which concatenates the first and second-order gradients to form the features, as an inexpensive solution to approximating high-pass filtering. Since RF can be used as a dictionary-learning-based tool, it inherits many properties from conventional dictionary-learning-based algorithms for feature extraction. However, the discriminative ability of those gradient-based features for random forest has been overlooked in the literature. We found, from experiments, that augmented features based on two gradient-magnitude filters can achieve more than 0.1dB quality improvement in RF-based SISR, with the same parameter setting.

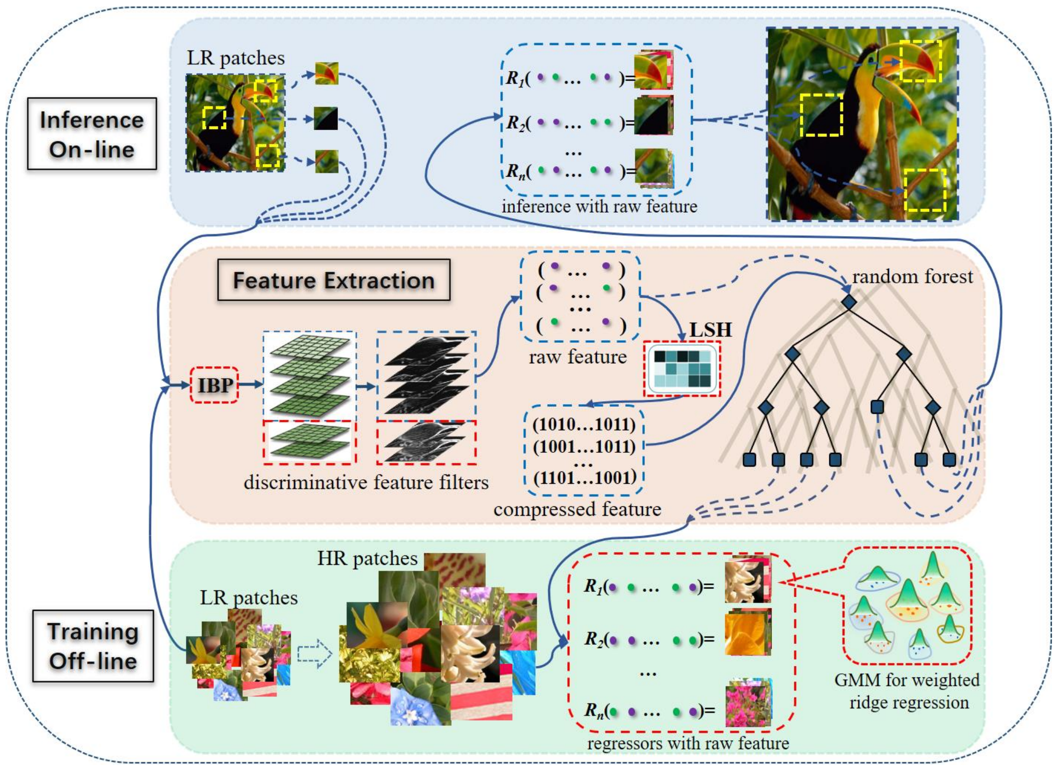


Fig. 6-1: An overview of the proposed FARF framework for image super-resolution. In the FARF-based image SR scheme, firstly, the initial coarse estimation is generated by using iterative back projection (IBP) instead of bicubic interpolation. More discriminative features are extracted by using the first and second-order gradients, as well as their magnitudes. Then, the conventional PCA is replaced by the generalized LSH for dimensionality reduction, and the compressed features are used for clustering at the split nodes in an RF. Finally, the respective regressors at the leaf-nodes are learned, by using the original high dimensional features with the GWRR models.

In most dictionary-learning-based algorithms, principal component analysis (PCA) is used for dimensionality reduction before classification and regression. The impact of using PCA has also been paid less attention in the literature. PCA projection may damage the structure of features, which are originally discriminative for clustering at the split nodes and regression at the leaf nodes. Motivated by the content-based image retrieval (CBIR) techniques in [101, 100], where the coarse-level search uses compressed features, while the fine-level search uses augmented features, we propose a similar approach for building more efficient trees, with their leaf nodes having better regression performances. To achieve this, our method uses the original features, rather than the compressed features generated by PCA as worked in [97, 62, 64, 65, 18, 88], so that more accurate regression and higher image quality improvement can be achieved. Moreover, unsupervised locality-sensitive hashing (LSH), instead of PCA, is employed for feature dimensionality reduction, which can reduce the damage on the feature structure after compression. The compressed features are used for clustering at the split nodes. Therefore, the resulting forest can improve the quality of reconstructed images. For the regression problems at the leaf nodes, in addition to using the augmented features without compression, we propose a generalized weighted ridge regression (GWRR) model as an extension of the work in [97]. The GWRR models are generated based on data distributions at the leaf nodes.

The main contribution of our method is on feature augmentation, so we call our method feature-augmented random forest (FARF). The pipeline of our FARF method, which includes feature extraction, the training stage, and inference stages for SISR, is shown in Fig. 6-1.

Having introduced the main idea of this chapter, the remainder of this chapter is organized as follows. In Section 2, we review the related works on SISR, particularly the RF-based approaches and our insights. In Section 3, we introduce the proposed FARF method, including the discriminative feature augmented by the gradient-magnitude filters, the generalized weighted ridge regression (GWRR) model, and the fine-tuned FARF. In Section 4, we evaluate our FARF scheme on public datasets, and conclusions are given in Section 5.

6.2 Image Super-Resolution via Random Forest

6.2.1 Conventional Patch-based Image Super-Resolution

Image SR attempts to achieve an impressive HR quality image from one or a set of LR images via artistic skills, which has been an active research topic for decades in the image restoration area. Generalized SR includes interpolation algorithms, such as the classic bicubic interpolation, and other edge-preserving algorithms [99, 60, 59, 58, 57].

The traditional super-resolution algorithms are based on pixel operations. Intuitively, operating on a "big pixel", i.e. a patch [105], is more effective. Since patch-based algorithms can preserve the local texture structure of an image, various methods based on image patches, such as non-local means [99], self-similarity [91], manifold learning [89], block-matching and 3D filtering (BM3D) [106], sparse representation [88], etc. have been proposed.

The neighbor-embedding (NE) methods [89, 90] are the milestone for patch-based dictionary learning methods. NE learns the mapping between low and high-resolution patches, with the use of manifold learning. Based on locally linear embedding (LLE), an LR patch can be represented as a linear combination of its nearest neighbors in a learned dictionary, and its HR counterpart can be approximated as a linear combination of the corresponding HR patches of its LR neighbors, with the same coefficients. Although the NE method is simple and sounds practical, a problem with the method is how to build an effective patch dictionary.

An approach to reducing the dictionary size is to learn a relatively smaller dictionary with discrete cosine transform (DCT) or wavelet fixed basis. However, the adaptiveness to data is sacrificed. In 2010, Yang *et al*. [88] proposed a sparse prior for dictionary learning. Using sparse coding, image representation can work with a relatively smaller dictionary, while keep the adaptiveness by learning the basis from data directly. This approach opens the era for sparse coding in the image inverse problems. Although the l^0-norm of α (the sparse coefficients) is an ideal regularization term for the sparse constraint, this strong constraint leads to an NP-hard problem in solving the coefficients α. Yang *et al*. [88] relaxed the l^0-norm to l^1-norm, so as to achieve a feasible solution.

Meanwhile, the effectiveness of sparsity is challenged [62, 5] by researchers, as to whether sparsity or collaborative representation really helps in image classification and restoration. As a natural solution to that, Timofte *et al*. proposed an anchored neighborhood regression (ANR) [62] framework, where there is no sparse constraint in the model. ANR replaces the sparse-decomposition optimization (l^1-norm) with a ridge regression (i.e. l^2-norm), where the coefficients can be computed offline and each coefficient can be stored as an atom (anchor)'s projection matrix in the dictionary. The offline learning is computationally intensive, but the online or prediction stage is very efficient. This approach has subsequently led to several variants. Timofte *et al*. later extended the ANR approach to the A+ [65]. In A+ [65], the coupled dictionaries are trained from a large pool of training samples (in the order of millions) rather than only from the anchoring atoms, which greatly improves the image quality. After that, more extensions based on ANR and A+ have emerged [97, 93, 95, 81].

6.2.2 Random-Forest-based Image Super-Resolution

Previously, in the above-mentioned dictionary-learning methods, the complexity of finding those similar patches by comparing an input patch with all the dictionary items has been overlooked. Recently, algorithms using random forest [62, 65, 8] have achieved state-of-the-art performances, in terms of both accuracy and efficiency for classification and regression tasks. This is mainly due to the use of ensemble learning and sublinear search based on binary decision trees. Schulter *et al*. [18] adopted random forest and the clustering-regression scheme to learn regressors from the patches in leaf nodes for SISR. With the same number of regressors, the RF-based algorithm can outperform or achieve comparable performance with A+ and its variants, in terms of accuracy but with less computational complexity.

RF-based image super-resolution, following a recent emerging stream [65, 91] on single-image SR, formulates the SR problem as a clustering-regression problem. These emerging approaches attempt to reconstruct an HR image from patches with the aid of an external database. These methods first decompose an image into patches, then classify the patches into different clusters, and later regressors are trained for all the respective clusters, which generate mappings from the features of an input LR patch to those of the corresponding HR patch. In the inference stage, an LR image follows the same procedures, such that it is divided into patches and features are extracted from each patch. Then, the patches are classified into different clusters using *K*-NN [18, 19] or RF [62, 65, 8], and their super-resolved HR patches are computed through regression in the leaf nodes (see Fig. 6-1). This kind of clustering-regression-based random forest [62, 65, 8] methods has achieved state-of-the-art performance in SISR, both in terms of accuracy and efficiency.

6.2.3 Deep Learning based Image Super-Resolution

In recent years, deep learning has achieved promising performances on image super-resolution [41, 98, 42, 43]. In [41, 98], milestone works on image super-resolution based on deep learning were presented, where a convolutional neural network (SRCNN) was proposed to learn an end-to-end mapping between LR and HR images for image super-resolution. Wang et al. [157] extended the SRCNN network to a sparse coding-based network by combining the domain knowledge. Later, a scheme with very deep networks for SISR was proposed in [42], where the convergence rate of the deep network is improved by using residual learning and extremely high learning rates. In addition, Ledig *et al*. [43] introduced a generative adversarial network (GAN)-based image super-resolution model (SRGAN), where the image perceptual loss function is reformulated as the combination of content loss and adversarial loss. Although deep-learning-based approaches have achieved promising progress on SISR, the heavy computational requirement is still a large burden even though the implementation is accelerated by GPU. This may limit them from those applications without powerful GPUs, such as smart mobile terminals.

6.3 Feature-Augmented Random Forest

Classification and regression can be regarded as probability problems from the statistical theory. Historical frequentist probability is the probability obtained from the relative frequency in a large number of trials. In contrast, the Bayesian probability is an interpretation of the concept of probability, in which probability is interpreted as an expectation taking the knowledge and personal belief into account. From the Bayesian theory, the posterior probability of a random event is a conditional probability, which can be calculated if the relevant evidence or context is considered. Therefore, the posterior probability is the probability $p(\theta|x)$ of the parameters θ given the evidence x. We denote the probability distribution function of the prior for parameters θ as $p(\theta)$, and the likelihood as $p(x|\theta)$, which is the probability of x given θ. Then, based on the Bayesian rule, the posterior probability can be defined as follows:

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{p(x)}. \tag{6.1}$$

The posterior probability can be written in a memorable form as:

$$Posterior\ probability \propto Likelihood \times Prior\ probability.$$

Based on the Bayesian framework, the likelihood term and the prior term are both required to be determined in order to solve the inverse problems, and the extracted features are normally worked as a prior or a likelihood, particularly on some image-restoration problems. From this point of view, most research works, from classic feature extractors to deep-learning neural networks, are essentially done under the Bayesian inference framework.

Since SISR is a well-known ill-posed problem, researchers have put their efforts into the priors of the problem with skills from mathematics, computer vision and machine learning. One of the obvious and most studied priors is the edge prior, which can be found in many pioneering works: new edge-directed interpolation (NEDI) [60], soft-decision adaptive interpolation (SAI) [59], directional filtering and data-fusion (DFDF) [58], modified edge-directed interpolation (MEDI) [57], and so on. The edge prior is effective on image processing, and the first and second-order gradients were studied and employed by Yang *et al*. [88] in a pioneering dictionary-learning-based algorithm. However, the effect of edge-based features has not been investigated in depth.

6.3.1 Augmented Features via Gradient Magnitude Filters

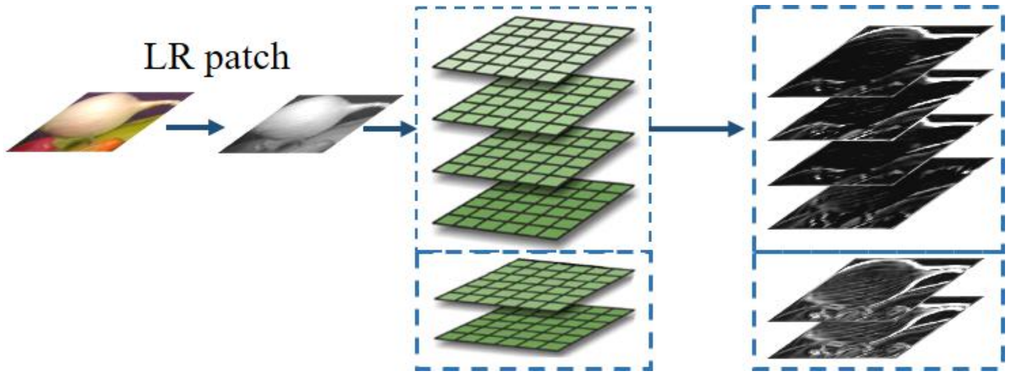


Fig. 6-2: Features extracted from an LR image through the first and second-order gradient filters in horizontal and vertical directions (the upper four), as well as the gradient magnitude filters (the below two), are concatenated to form augmented features with more discriminative properties.

For the clustering and classification problems, feature engineering is a critical research point, and in some cases, the chosen features may dominate the performance. A feature filter F, whose coefficients are computed to fit the most relevant parts in the LR image patches, is employed, and the generated features can achieve more accurate predictions for reconstructing their corresponding HR image patches, as shown in Fig. 6-2.

Normally, it is unstable to directly use pixel intensities as features, which are susceptible to the environmental lighting variations and camera noise. Instead, the differences between the neighboring pixels' intensity values, which are computationally efficient, and are immune to lighting changes and noise, are examined. This type of feature can be implemented efficiently through convolutional filters. Typically, the feature filter F can be chosen as a high-pass filter, while in [62, 64, 65, 88], the first and second-order gradient operators are used to generate an up-sampled version from a low-resolution image, then four patches are extracted from the gradient maps at same location, and finally the patches are concatenated to form feature vectors. The four 1-D filters used to extract the gradients are described in Eqn. (6.2),

$$\left.\begin{array}{ll} F_1 = [-1, 0, 1], & F_2 = {F_1}^T \\ F_3 = [1, 0, -2, 0, 1], & F_4 = {F_3}^T \end{array}\right\}, \tag{6.2}$$

where F_1 and F_2 are first-order gradient filters, while F_3 and F_4 are second-order gradient filters.

These features can work well on dictionary-learning-based methods, because when searching a matched patch in a dictionary, the distance is calculated based on the whole feature vectors with the Euclidean distance. However, when training a split node in a decision tree of an RF, only one or a few of the features are chosen as candidate features for comparison. Therefore, more discriminative features are required for RF, when compared with dictionary-learning-based methods.

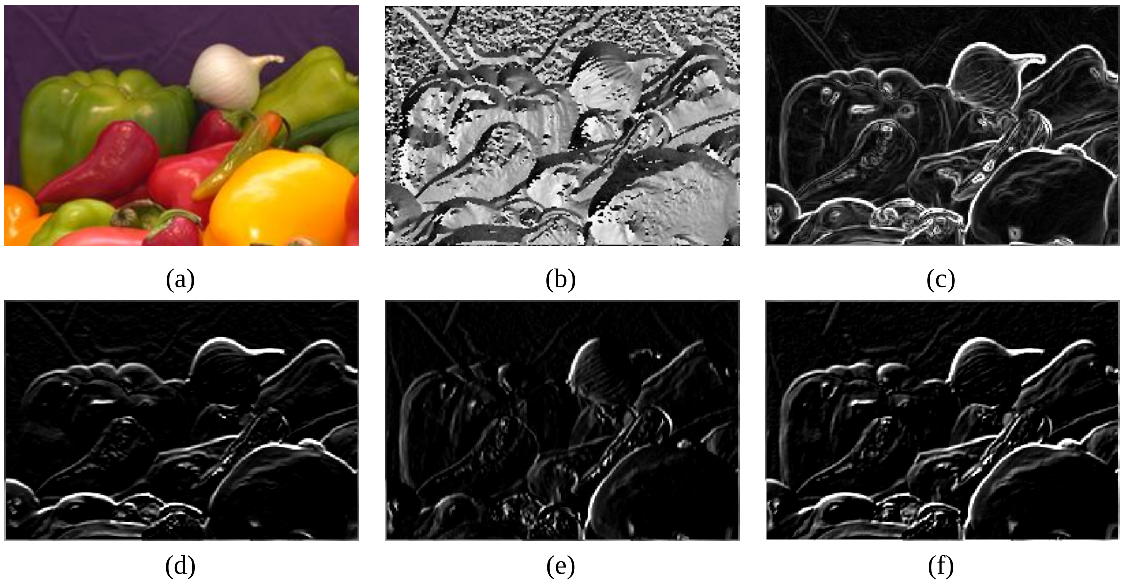


Fig. 6-3: Visualization of the features from a natural image: (a) original color image, (b) image *gradient orientation*, (c) image *gradient magnitude*; (d) *horizontal gradient* $\partial I/\partial x$, (e) vertical *gradient* $\partial I/\partial y$, (f) the sum: $(\partial I/\partial x + \partial I/\partial y)$.

The first and second-order *gradients* of an image can provide the directions of edges in a perceptual manner, as shown in Fig. 6-3, which can be calculated as follows:

$$\nabla I = \left[\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right]^T, \tag{6.3}$$

where $\partial I/\partial x$ and $\partial I/\partial y$ are the *gradients* in the *x*-axis and *y*-axis directions, respectively, at a given pixel. Meanwhile, the *gradient magnitude* image can provide the edge strength, as described in Eqn. (6.4).

$$\|\nabla I\| = \sqrt{(\frac{\partial I}{\partial x})^2 + (\frac{\partial I}{\partial y})^2}. \tag{6.4}$$

With the natural image shown in Fig. 6-3, it can be observed that the *gradient magnitude* image has more detailed textures than the *gradient* images ($\partial I/\partial x$ and $\partial I/\partial y$), as well as the sum of the horizontal *gradient* and vertical *gradient* image, i.e. $\partial I/\partial x + \partial I/\partial y$, perceptually. An explanation for this phenomenon is that non-linear features are usually more discriminative. Thus, in our work, all the first and second-order *gradients,* and *gradient magnitude* are employed, and are concatenated to form more discriminative, augmented features.

On the other hand, the image *orientation* (gradient angle) is defined by the following formulation,

$$\angle \nabla I = \tan^{-1}(\partial y/\partial x)\,, \tag{6.5}$$

where $\tan^{-1}(.)$ is the gradient orientation, with a value between -90° and 90°. As shown in Eqn. (6.5), when the value of ∂x is equal to 0 or close to 0, the value of $\angle\nabla$ becomes infinitely large and unstable, i.e., different ∂y will result in approximately same $\angle\nabla$ value.

Based on this analysis, we only use the two *gradient magnitude* filters derived from the four gradient filters [88], shown in (2), to generate the augmented features. Experiments validate that the use of the augmented features can improve the conventional

RF algorithm [18] to achieve a performance gain of more than 0.1dB, which is a remarkable improvement, with the same setting and parameters.

6.3.2 Fine-grained Features for Regression

The inference stage of the RF-based image super-resolution process is similar to the content-based image retrieval (CBIR) framework, as shown in Fig. 6-4. The general approximated nearest neighbor (ANN) search framework [101, 100] is an efficient strategy for large-scale image retrieval, which mainly consists of 4 parts: (1) extracting compact features (e.g., the locality-sensitive Hashing (LSH) [102] features) for a query image; (2) coarse-level search using Hamming distance to measure the similarity between binary compact Hash features, then narrow the search scope into a smaller candidate group; (3) fine-level search by using Euclidean distance to measure the similarity between their corresponding feature vectors; and (4) finding the object in the smaller candidate group that is the nearest one to the query image.

Locality sensitive hashing (LSH) [158, 159] is a series of algorithms which are used for handling largescale data processing with high dimensionality. The main idea behind LSH is to formulate a family of functions which can map (hash) high dimensional features into buckets so that similar features can fall into the same bucket with a high probability.

In the inference stage of conventional RF-based SISR, PCA projection is worked as a hash-like function to reduce the feature dimension for decreasing the search range, which can speed up the searching as the coarse-level search in a CBIR framework. However, the impact of using PCA on feature dimensionality reduction has been overlooked in previous works [97, 62, 64, 65, 18, 88]. In our algorithm, the LSH based scheme from [108] is employed to further improve feature discrimination. This LSH based scheme [108] transforms the original feature space into a compactly pre-clustered feature space, via a trained rotation matrix. The additional load of the rotation in the inference stage is ignorable, and the feature dimension will keep the same as PCA does [97, 62, 64, 65, 18, 88], i.e., for an enlargement with scale factor 3, the original dimension is 486 (for the 9×9 size image patch with 6 filters), and the final dimension is about 50 after 90% dimensionality reduction with PCA or LSH.

Inspired by the fine-level search using augmented features in CBIR frameworks, the high-dimensional features used in the leaf nodes of an RF can further improve the prediction accuracy in the regression step, which has not been studied previously. Consequently, we use the original features, rather than the PCA or LSH compressed features, to perform ridge regression in the leaf nodes. Experimental results show that this new RF scheme, by using this augmented feature, can greatly improve the quality of super-resolved images. Another explanation for this is that the regression problems can benefit more from the higher dimensional features than classification problems.

Based on the observation that the original edge-like features are used for the final regressors in the leaf nodes and the compressed features (either produced by PCA or LSH) are used for clustering in the split nodes, a new clustering-regression-based SISR approach can be designed as shown in Fig. 6-4. In this new scheme, the original-compressed coupled feature sets are worked for different purposes at different stages, i.e., the original edge features are used for regression in the leaf nodes, and the compressed features derived from the LSH-like functions are employed for node splitting (clustering) in the training stage, and node searching in the inference stage in the split nodes. On training the split nodes, the entropy is replaced by variance as the works in [18, 108].

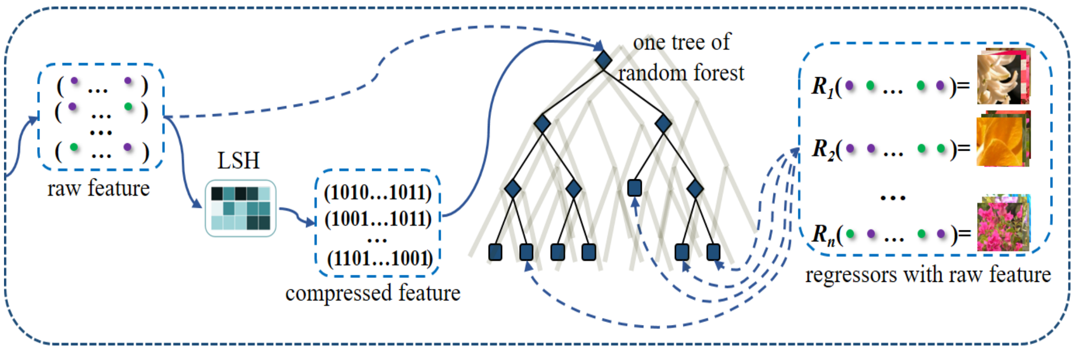


Fig. 6-4: Augmented features for regressors and the LSH compressed features for searching in the trees of a random forest.

In the new scheme, we unify the research of LSH-based SISR and image retrieval (CBIR) [100, 101]. In brief, the new achievement on unsupervised LSH can be evaluated not only in CBIR systems, but also in the clustering-regression RF-based SISR methods. Moreover, as evidence from [108], using proper unsupervised LSH models, e.g., iterative quantization (ITQ) [61], for feature-dimension reduction, instead of PCA, can reduce the damage on the image structure. This can further improve the super-resolved image quality. Different from [108] using an ITQ-like algorithm to rotate the original features into a new feature space, with the use of the proposed original-compressed coupled feature sets, any unsupervised LSH generated features can directly be employed.

6.3.3 Generalized Weighted Ridge Regression Model

In this sub-section, we further analyze the ridge regression employed in the RF leaf nodes. The anchored neighborhood regression (ANR) [62] model relaxes the l^1-norm to the l^2-norm, with least-squares minimization as the following equation,

$$\min_{\alpha} \|F\mathrm{D}_l\alpha - Fy\|_2^2 + \lambda\|\alpha\|_2, \tag{6.6}$$

where α is the coefficients, and F is a feature-extraction operator on the LR patches, which aims to extract discriminative features from LR patches, rather than using the raw pixel intensity. In our formulation, D_l and D_h represent the low and high-resolution coupled dictionaries trained jointly from LR and HR patch samples, respectively. Based on the ridge regression [76] theory, this l^2-norm constrained least square regression regularized problem has a closed-form solution, according to the Tikhonov regularization theory, as follows:

$$\alpha = (D_l^T D_l + \lambda I)^{-1} D_l^T F y. \tag{6.7}$$

With the assumption in [88], where HR patches and their counterpart LR patches share the same reconstructed coefficient α, i.e. $x = \mathrm{D}_h \alpha$, from Eqn. (6.7) we have

$$x = \mathrm{D}_h (D_l^T D_l + \lambda I)^{-1} D_l^T F y. \tag{6.8}$$

If we define P_G as a pre-calculated projection matrix, as follows,

$$P_G = \mathrm{D}_h (D_l^T D_l + \lambda I)^{-1} D_l^T, \tag{6.9}$$

then the HR patches can be reconstructed with $x = P_G F y$.

Having studied the model in Eqn. (6.8), the authors in [97] argued that different weights should be given to different atoms when reconstructing an HR patch so as to emphasize the similarity to the anchor atom. Based on this idea, authors in [97] proposed a weighted collaborative representation (WCR) model by generalizing the normal collaborative representation (CR) model in ANR, as follows:

$$\min_{\alpha} \|F\mathrm{D}_l \alpha - F y\|_2^2 + \|\lambda_{WCR} \alpha\|_2, \tag{6.10}$$

where λ_{WCR} is a diagonal weight matrix, whose non-zero diagonal entries are proportional to the similarities between the atoms and the anchor atom.

Same as the ANR model, a new closed-form solution can be computed offline through the following equation,

$$\alpha^* = (D_l^T D_l + \lambda_{WCR})^{-1} D_l^T F y, \tag{6.11}$$

and the new projection matrix can be derived as

$$P_G^* = \mathrm{D}_h (D_l^T D_l + \lambda_{WCR})^{-1} D_l^T. \tag{6.12}$$

The WCR model further improves the ANR/A+ model in terms of image quality, while keeping the same level of computation. In [82], the local geometry prior of the data sub-space is used. However, all the weighted ridge regression models [97, 82] are constructed based on an existing dictionary, e.g., Zeyde *et al*. [64] used K-SVD to train a sparse-coding-based dictionary with 1,024 items.

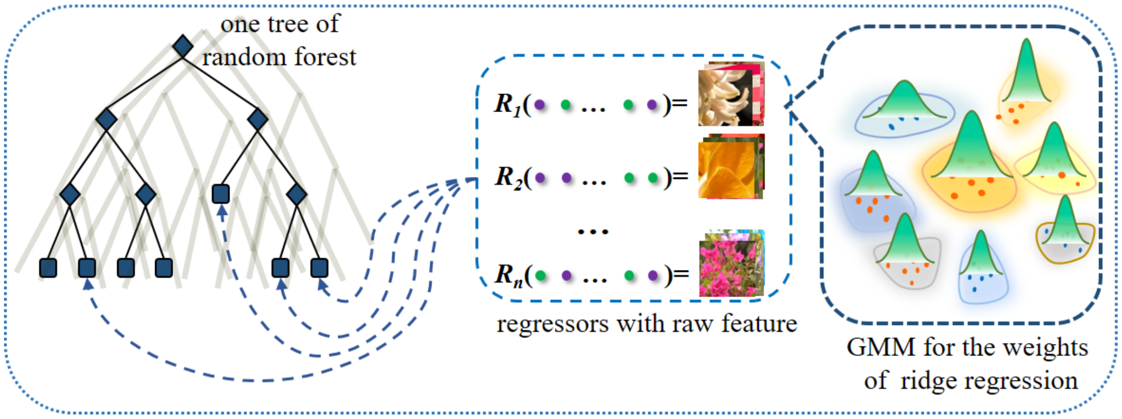

one tree of random forest
$R_1(\bullet \bullet \ldots \bullet \bullet)=$
$R_2(\bullet \bullet \ldots \bullet \bullet)=$
...
$R_n(\bullet \bullet \ldots \bullet \bullet)=$
regressors with raw feature
GMM for the weights of ridge regression

Fig. 6-5: Gaussian mixture model (GMM) is used to generate the weights for weighted ridge regression, and the weight of each entry lies on its belonging cluster's size and its centrality in the belonging cluster.

When training the regressors in an RF, there is no existing anchor point in the clustered groups of the leaf nodes, similar to the previous models [97, 82]. A solution to the mentioned problem is inspired by the work on image classification using locality-constrained linear coding (LLC) [103], where the Gaussian mixture model (GMM) is used to describe the locality-constrained affine subspace coding (LASC) [104]. We employ GMM to construct the data distribution in the sub-space for each leaf node, which derives the weights of all the entries in the ridge regression models. Through the derived weights, we can obtain a generalized weighted ridge regression (GWRR) model for ridge regression. The new projection matrix is given as follows:

$$P_G^* = \mathrm{D}_h(D_l^T D_l + \lambda_{GWRR})^{-1} D_l^T, \tag{6.13}$$

where λ_{GWRR} is a diagonal weight matrix, and diagonal entries are the weights of samples in the leaf-node. Each sample's weight is related to its belonging cluster's size and its local centrality in its belonging cluster, as illustrated on the right of Fig. 6-5. In other words, the cluster's weight is proportional to its sample amount, and the local weight of a sample is depending on how close it is to the cluster center. Obviously, a query input, falling into a bigger cluster and closer to the center of the belonging cluster, achieves a larger weight. In a rough form, the diagonal weight matrix λ_{GWRR} is given as follows:

$$\lambda_{GWRR} = diag\{[w_1; w_2; \ldots; w_i; \ldots]\}, w_i \propto C_i^k \times d_i^k, k = (1, \ldots, K) \tag{6.14}$$

where w_i is the weight of the i^{th} entry, m is the number of samples in the leaf nodes, C_i^k is the k^{th} cluster's weight for the i^{th} entry, d_i^k is the i^{th} entry's local weight in the k^{th} cluster, which is approximated as the inverse of the distance to the center of the belonging cluster, and K is the number of clusters generated by the GMM model for a leaf node.

Experimental results in Table-6.1 show that the proposed GWRR model can achieve the same level of performance as WCR [97] and obtain 0.2dB gain than the ANR [62] model. Note that when the number of samples in a leaf node becomes bigger, the performance of the GWRR model will achieve less advantages than the normal regression model, because the higher weights will be averaged by a large number of other samples. Theoretically, the regression of a leaf node can benefit from the GWRR model, particularly when there are a few samples falling into the leaf nodes.

Images	*baboon*	*baby*	*bird*	*butterfly*	*foreman*	*head*	*lenna*	*woman*	***Average***
ANR	23.52	35.06	34.44	25.74	32.92	33.54	32.92	30.17	31.04
WCR	23.55	35.09	34.75	26.18	33.51	33.61	33.16	30.42	**31.28**
GWRR	23.54	35.09	34.74	26.13	33.46	33.58	33.12	30.38	31.25

Table-6.1: Performances of ANR [62], WCR [97], and the proposed GWRR, in terms of PSNR (dB) with an upscale factor (×3) on some public standard test images.

6.3.4 Iterative Back Projection for Initial Coarse Estimation

In a broad sense, SISR is a low-level computer vision task, which attempts to restore an HR image $\mathcal{X}$ from a single input LR image $\mathcal{Y}$. A mathematical model for image degradation can be formulated as follows:

$$\mathcal{Y} = (\mathcal{X} * \mathcal{B}) \downarrow s, \tag{6.15}$$

where $\mathcal{B}$ is a low-pass (blur) filter and $\downarrow s$ denotes the down-sampling operator with a factor of s. Based on a given LR image $\mathcal{Y}$, how to achieve an approximated HR image $\hat{\mathcal{X}}$ is a classic inverse problem, which requires priors based on the Bayesian theory.

Irani and Peleg [107] firstly proposed an iterative back projection (IBP) method for SR reconstruction. IBP is an effective way to obtain an HR image when comparing it with other SR methods. In the IBP method, the reconstruction error of an estimated LR image $\hat{\mathcal{Y}}$ is the difference between the input LR $\mathcal{Y}$ and the synthesized image $\hat{\mathcal{Y}}$ generated from the estimated HR image $\hat{\mathcal{X}}$ as follows:

$$e(\hat{\mathcal{Y}}) = \mathcal{Y} - \hat{\mathcal{Y}} = \mathcal{Y} - (\hat{\mathcal{X}} * \mathcal{B}) \downarrow s. \tag{6.16}$$

IBP can efficiently obtain the HR image by minimizing the reconstruction error defined in Eqn. (6.16). For the IBP approach on SISR, the updating procedure can be summarized as the following two steps, performed iteratively:

- Compute the reconstruction error $e(\hat{\mathcal{X}})$ with the following equation:

$$e(\hat{\mathcal{X}}) = e(\hat{\mathcal{Y}}) \uparrow \mathrm{s} * p, \tag{6.17}$$

where $\uparrow$ is the up-sampling operator and p is a constant back-projection kernel to approximate the inverse operation of the low-pass filter $\mathcal{B}$.

- Update the estimated HR image $\widehat{\mathcal{X}}$ by back-projecting errors as follows:

$$\widehat{\mathcal{X}}^{t+1} = \widehat{\mathcal{X}}^{t} + e\left(\widehat{\mathcal{X}}^{t}\right), \tag{6.18}$$

where $\widehat{\mathcal{X}}^{t}$ is the estimated HR image at the t-th iteration.

Most learning-based algorithms [97, 62, 64, 5] follow the milestone work in [88], which uses the coarse estimation firstly obtained via bicubic interpolation. As we know, the classic IBP algorithm is an efficient way to obtain high-quality up-scaled images, but it will inevitably produce artifacts (such as ringing, jaggy effects, and noise) at the output, because the kernel operator p in Eqn. (6.17) is hard to estimate accurately. That is the reason why algorithms with IBP need an additional denoising process [109, 23, 107]. However, the sparse-constraint-based approach [88] does not have this denoising capability.

As the l^2-norm constraint-based ridge regression has the denoising effect, due to its averaging-like process, this means that the ridge regression-based RF scheme has the denoise capability intrinsically. Based on this observation, we obtain the coarse estimation of an HR image $\widehat{\mathcal{X}}$ by applying IBP to the corresponding input LR image $\mathcal{Y}$. Experimental results in Table-6.2 and Table-6.3 validate that using IBP, instead of bicubic, to obtain the initial coarse estimation can help the RF-based SR method obtain a remarkable improvement.

6.3.5 Fine-Tuning with Proper Trees in Random Forest

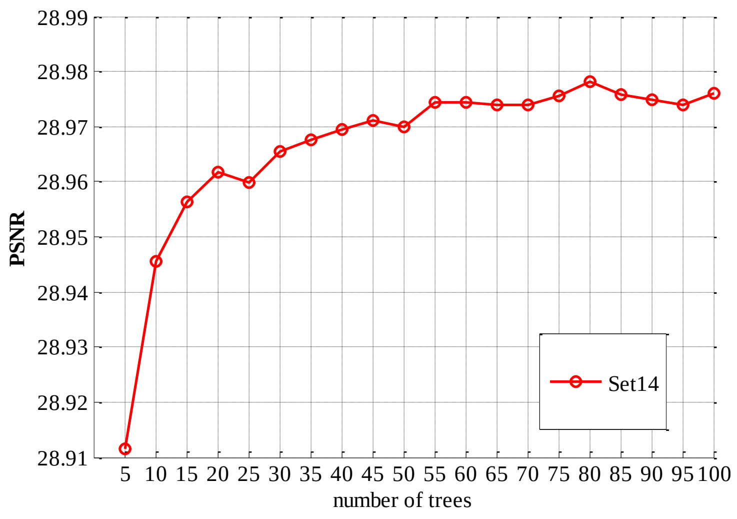


Fig. 6-6: The image super-resolution quality, in terms of PSNR, with different numbers of trees in a random forest for super-resolution (3x) experiments on Set14. The number of trees = 45 gives a better trade-off between efficiency and complexity.

As the number of trees is an important parameter in RF-based approaches, we plot the performance with respect to the number of trees. As shown in Fig. 6-6, the performance of the RF-based image super-resolution method increases as expected, but the increment becomes relatively smaller after a certain number of trees are used. The experimental results in Fig. 6-6 were obtained on the Set14 dataset, and 2 million samples from the dataset were used for all training stages. It shows that using 45 trees is an optimal number, as a tradeoff between performance and computational cost. Therefore, we set the number of trees for the proposed FARF method at 45, and our method with this number is denoted as FARF*. The performances of our methods, and other methods, are tabulated in Table-6.2 and Table-6.3. We also compare our methods with a recently proposed deep-learning-based algorithm, SRCNN algorithm [41, 98] and SCN algorithm [157], and our methods outperform them in some cases.

6.3.6 Algorithm Workflow

The training and inference stages of the proposed FARF algorithm are described in Algorithm 6.1 and Algorithm 6.2, respectively. To help the readers understand this chapter, the source code of our algorithm will be available at: https://github.com/HarleyHK/FARF, for reference.

```
Algorithm 6.1: Training Stage of FARF-based Image Super-Resolution:
Input: ${\mathcal{y}^i, \mathcal{x}^i}_{i=1}^N$: training LR-HR patch pairs;
Output: the trained random forest $\mathcal{T}$ with regressors $\mathcal{R} = (\mathcal{R}_1, \ldots)$, the LSH model: $\mathcal{M}_{LSH}$;
1: Upscale LR patch images as initial coarse estimations using IBP;    ⇒ {Eqns. (6.17, 6.18)}
2: Obtain the features calculated from LR image by using the first and second-order gradient filters,
   and the gradient magnitudes on the up-scaled coarse versions;    ⇒ {Eqns. (6.3, 6.4)}
3: Conduct LSH on the raw features to obtain compressed features, at the same time obtain the
   trained LSH projection model $\mathcal{M}_{LSH}$;
4: Train a random forest with the compressed features via the LSH model $\mathcal{M}_{LSH}$;
5: Train the weighted ridge regressors $\mathcal{R}$ by the GWRR models in leaf nodes; ⇒ {Eqn. (6.13)}
6: Save the random forest $\mathcal{T}$ with ridge regressors $\mathcal{R}$ , and the trained LSH model $\mathcal{M}_{LSH}$.
```

```
Algorithm 6.2: Inference Stage of FARF-based Image Super-Resolution:
Input: testing LR image $\mathcal{Y}$, the trained random forest $\mathcal{T}$ with ridge regressors $\mathcal{R} = (\mathcal{R}_1, \ldots)$, the
   trained LSH model $\mathcal{M}_{LSH}$;
Output: super-solved image $\hat{\mathcal{X}}$;
1: Upscale patches from LR $\mathcal{Y}$ to form an initial coarse estimation by IBP;  ⇒ {Eqn. (6.17, 6.18)}
2: Compute the discriminative features for all the patches;    ⇒ {Eqns. (6.3, 6.4)}
3: Compute the compressed feature with the LSH model $\mathcal{M}_{LSH}$;
4: For each patch, using the compressed feature to search in the leaf nodes to obtain its
   corresponding regressor from the trained random forest $\mathcal{T}$;
5: Compute the super-resolved image $\hat{\mathcal{X}}$ through all the super-solved patches by weighted ridge
   regressors $\mathcal{R}$ in leaf nodes.    ⇒ {Eqn. (6.10)}
```

6.4 Experiments

In this sub-section, we evaluate our algorithm on standard super-resolution benchmarks Set 5, Set14 and B100 [80], and compare it with some state-of-the-art methods. They are *bicubic* interpolation, adjusted anchored neighborhood regression (A+) [65], standard RF [18], alternating regression forests (ARF) [18], the convolutional neural-network-based image super-resolution (SRCNN) [41, 98], the naive Bayes super-resolution forest (NBSRF) [19] and the sparse coding based network (SCN) [157], as listed in Table-6.2 and Table-6.3. We set the same parameters for all the RF-based algorithms, i.e., the number of trees in an RF is 10, and the maximum depth of each tree is 15. We use the same set of training images (91 images) for all the algorithms, as previous works [62, 64, 65, 8] do. RF+ means a normal RF-based algorithm added with the two *gradient magnitudes* augmented features, and RF# is the normal RF-based algorithm, where the original raw features, instead of using the PCA compressed features, are used to learn the regressors in leaf nodes. FARF$^-$ denotes trimmed version of our proposed feature-augmented RF scheme, which combines RF+ and RF# by adding the gradient magnitude features and using the original raw features for regression. FARF means the normal version or our proposed algorithm, which is based on FARF$^-$, the superior, unsupervised LSH projection [108], instead of PCA, is employed for dimensionality reduction, and the generalized weighted ridge regression (GWRR) model is used for the leaf-nodes' regressors. FARF* is a further refined version of FARF, by performing further fine-tuning: (1) employing IBP, instead of the traditional *bicubic* interpolation algorithm, to obtain the initial coarse estimation in the inference stage, and (2) setting the proper number of trees (e.g., 45) for training an RF.

dataset	#	*bicubic*	A+	RF	ARF	RF^{+}	RF$^{\#}$
Set5	×2	33.66/0.00	36.55/0.51	36.52/0.03	36.65/0.82	36.67/0.04	36.63/0.05
	×3	30.39/0.00	32.59/0.33	32.44/0.04	32.53/0.62	32.56/0.05	32.53/0.05
	×4	28.42/0.00	30.29/0.23	30.10/0.05	30.17/0.71	30.18/0.06	30.22/0.06
Set14	×2	30.23/0.00	32.28/1.11	32.26/0.05	32.33/1.43	32.37/0.06	32.32/0.07
	×3	27.54/0.00	29.13/0.66	29.04/0.07	29.10/1.12	29.17/0.08	29.11/0.09
	×4	26.00/0.00	27.33/0.47	27.22/0.08	27.28/0.91	27.31/0.09	27.29/0.09
B100	×2	29.32/0.00	30.78/0.00	31.13/0.06	31.21/1.52	31.22/0.07	31.23/0.07
	×3	27.15/0.00	28.18/0.00	28.21/0.07	28.26/1.23	28.27/0.08	28.26/0.08
	×4	25.92/0.00	26.77/0.00	26.74/0.07	26.77/1.02	26.78/0.09	26.79/0.09

dataset	#	FARF^{-}	FARF	FARF*	SRCNN	NBSRF	SCN
Set5	×2	36.68/0.06	36.78/1.03	**36.81**/3.02	36.66/3.21	36.76/0.04	36.58/5.12
	×3	32.62/0.06	32.73/1.12	**32.78**/3.13	32.75/3.36	32.75/0.05	32.68/5.31
	×4	30.27/0.07	30.39/1.20	30.45/3.18	**30.48**/3.11	30.44/0.06	30.41/5.36
Set14	×2	32.37/0.08	32.41/1.15	**32.45**/3.23	32.42/6.52	**32.45**/0.06	32.35/6.02
	×3	29.17/1.02	29.23/1.24	**29.29**/3.27	29.28/6.41	29.25/0.08	29.16/6.13
	×4	27.36/1.10	27.45/1.31	27.48/3.35	**27.49**/6.38	27.42/0.09	27.39/6.21
B100	×2	31.34/0.92	31.35/1.22	**31.38**/3.28	31.36/8.24	31.36/0.09	**31.38**/6.16
	×3	28.30/1.05	28.35/1.25	28.38/3.32	**28.41**/8.37	28.39/1.02	28.33/6.30
	×4	26.83/1.20	26.88/1.41	**26.91**/3.46	26.90/8.42	26.89/1.13	26.88/6.51

Table-6.2: Results of proposed method compared with state-of-the-art works on 3 datasets in terms of PSNR (dB) and runtime (seconds) using three magnification factors (#) (×2, ×3, ×4).

Table-6.2 summarizes the performances of our proposed algorithms on the 3 datasets, in terms of the average peak signal to noise ratio (PSNR) scores and runtime (seconds), with different magnification factors (×2, ×3, ×4). Table-6.3 gives more details of the results on some images from the Set5 dataset, with magnification factor ×3. As the results have shown based on the 3 datasets, our proposed algorithm FARF outperforms A+ and ARF for all the magnification factors.

Set5(×3)	*bicubic*	Zeyde	A+	RF	ARF	FARF^{-}	FARF	FARF*	SRCNN	NBSRF	SCN
baby	33.91	35.13	35.23	35.25	35.15	35.20	35.34	**35.37**	35.25	35.28	35.25
bird	32.58	34.62	35.53	35.23	35.31	35.39	35.53	**35.54**	35.47	35.49	35.42
butterfly	24.04	25.93	27.13	27.00	27.39	27.65	27.68	27.82	**27.95**	27.87	27.78
head	32.88	33.61	33.82	33.73	33.73	33.75	33.84	**33.85**	33.71	33.78	33.65
woman	28.56	30.32	31.24	30.98	31.08	31.11	31.27	31.34	**31.37**	31.33	31.28
average	30.39	31.92	32.59	32.44	32.53	32.62	32.73	**32.78**	32.75	32.75	32.68

Table-6.3: Results of the proposed method compared with state-of-the-arts methods on 3 datasets in terms of PSNR (dB) with magnification factors (×3) on dataset Set5.

The objective quality metric, PSNR, in Table-6.2 also shows that the fine-tuned FARF, i.e. FARF*, can further improve the image quality, which is comparable to recently proposed state-of-the-art deep-learning-based algorithms, such as SRCNN [41, 98] and SCN [157]. Comparing our proposed FARF algorithm to other methods, the improved visual quality of our results is obvious, as shown in Fig. 6-7. This shows that our method can produce more details, particularly on some texture-rich regions.

6.5 Conclusions

This chapter presents a feature-augmented random forest (FARF) scheme for the single-image super-resolution (SISR) task by augmenting features and redesigning the inner structure of a random forest (RF), with different feature recipes used at different stages, where the compressed features are used for clustering in the split nodes and the original features are used for regression in the leaf nodes. The contributions of this chapter are threefold: (1) the more discriminative gradient magnitude-based augmented features are proposed for clustering in split nodes and regression in leaf nodes; (2) By replacing principal component analysis (PCA) with a generalized unsupervised locality-sensitive hashing (LSH) model for dimensionality reduction, we lay out an original-compressed coupled feature set for tackling the clustering-regression tasks, and unify SISR and content-based image retrieval for LSH performance evaluation; and (3) we have extended the weighted collaborative representation model to a generalized weighted ridge regression model for ridge regression. The proposed FARF scheme can achieve highly competitive quality results, e.g., obtaining about a 0.3dB gain in PSNR, on average, when compared to the conventional RF-based super-resolution approaches. Furthermore, a fine-tuned version of our proposed FARF approach is provided, whose performance is comparable to recent state-of-the-art deep-learning-based algorithms.

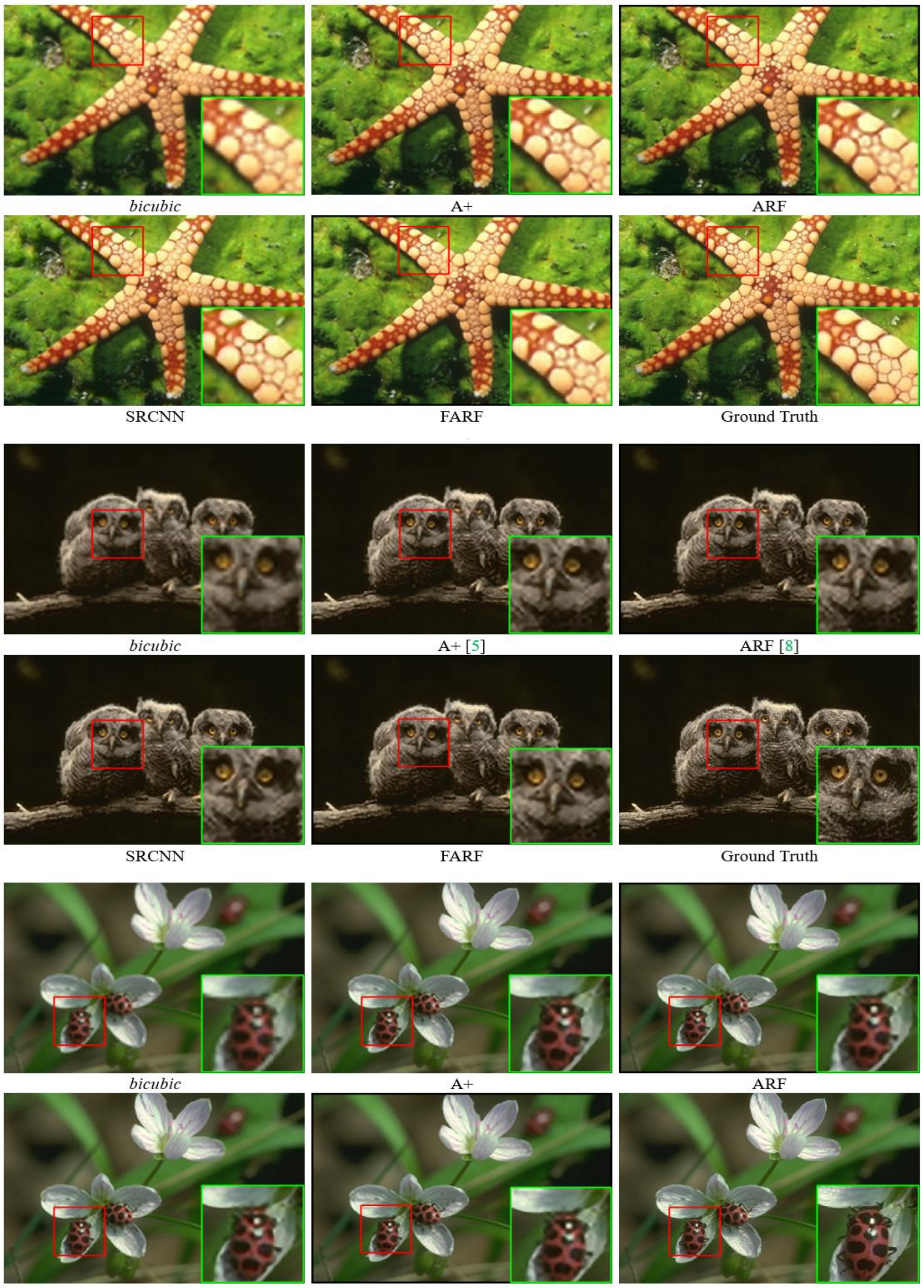


Fig. 6-8: Super-resolved (×3) images from B100, *bicubic*, A+ (ACCV-2014) [65], ARF (CVPR-2015) [18], SRCNN (PAMI-2016) [98], our proposed FARF algorithm, and ground truth. The results show that our FARF algorithm can produce more details and its performance is comparable to a recent state-of-the-art deep-learning method [98].

Chapter 7. Conclusions and Future work

7.1 Summary and Conclusions

In this thesis, we first introduced the concept and development of random-forest technologies, as well as random-forest-based applications. Some of the existing challenges, which serve as the motivation for this research work, are discussed in detail. Then, we give literature reviews on three main topics: random forest, face alignment and image super-resolution in Chapter 2. We have presented a novel and efficient shape-index-derived feature, which can work with random-forest-based regressors for face alignment in Chapter 3. We have further designed a new scheme, which can combine two schools of face-alignment approaches for face alignment in Chapter 4. In Chapter 5, we designed a scheme to rotate the feature space into a more compact feature space, which can help the random forest on the classification and regression tasks, and the variant random-forest model can also be employed on image super-resolution. In Chapter 6, we presented a feature-augmented random-forest (FARF) scheme for single image super-resolution, through the work on feature engineering in random forest. In this final chapter, in addition to summarizing the main contributions of this research, we also discuss some possible directions for future work.

7.2 Future Research Work

This thesis, which has presented a few new ideas and methods, is just a snapshot of our ongoing research, undertaken in the field of random-forest-based computer-vision applications. In this section, some possible directions for our future research will be discussed. This future research may be carried out in the following fields:

(a). The high-dimension problem: As the random forest has the capability of handling high-dimension data, researchers have often used it for high-dimensional data analysis. However, when the data dimension becomes higher, the computation and accuracy of classification and regression will be challenged. Principal component analysis (PCA) is a conventional method for data-dimensional reduction, but PCA may destroy the data structure. Although this thesis has proposed using locality-sensitive hashing (LSH) to replace PCA for feature-dimensionality reduction, an algorithm, which can help random forest to reduce data dimension with promising performance, is still an open research topic.

(b). The parallel speed-up problem: The more trees there are in a random forest, the better the performance of a random forest will be. To speed up a random-forest-based model on CPU or GPU instructions, in parallel, will be a significant work for random-forest applications.

(c). The collaboration problem: Currently, each tree in a random forest works independently as an individual expert, which means that there is no collaboration between them. To let the trees in a forest work as a team, with collaboration between them, for better prediction performance, would be an interesting research topic.

(d). The parameter-setting problem: Currently, the main parameters in a random forest, such as the depth of a tree and the maximum sample number in a leaf node, are set by experimental results, therefore optimal parameters derived by theory would be a research topic that would enhance random-forest performance.

(e). The integration with deep learning problem: These days, deep learning has shown its strength in a lot of computer-vision applications because of its strong learning capability. As random forest is a classic ensemble tool, combining random forest with deep learning would be an interesting research topic in this area. This integration can bring some new variants of random forest, e.g., the deep forest [111] or the deep neural decision forest [112]. In [111], the authors proposed a novel approach to enhance the classification capability of the trees in a random forest by the representation learning ability of deep (neural) networks, and this scheme can work in an end-to-end trainable architecture. Again in [112], the authors presented the gcForest model, which is a decision-tree ensemble scheme, which is highly competitive, compared to current deep neural networks. Also, the deep reinforcement learning has emerged as a hot research topic after Google’s AlphaGo [116] defeated Go masters, in a win for artificial intelligence (AI). Recent years have shown the achievements of deep reinforcement learning (DRL) techniques on tasks with visual observations, such as Atari games [114] and path planning [115], and in particular, when Google DeepMind's AlphaGo algorithm [116] beat world class Go players in 2016. In those DRL-based game algorithms, including the AlphaGo algorithm, the classic Monte-Carlo Tree Search (MCTS) algorithm [113] has been widely employed in computer games. As the MCTS algorithm employs decision trees to estimate the expected result and the reinforcement learning takes a "trial and error" learning process to learn the strategies from states to actions, which can benefit the random decisions in random forests.

References

[1] S. Ren, X. Cao, Y. Wei, and J. Sun, "Face alignment at 3000 fps via regressing local binary features," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 1685-1692.

[2] H. Zhou, K.-M. Lam, and X. He, "Shape-appearance-correlated active appearance model," Pattern Recognition (PR), vol. 56, pp. 88-99, 2016.

[3] T. F. Cootes and C. J. Taylor, "Active shape models — 'smart snakes'," in BMVC92: Springer, 1992, pp. 266-275.

[4] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearance models," in European conference on computer vision (ECCV), 1998, pp. 484-498: Springer.

[5] M. Dantone, J. Gall, G. Fanelli, and L. Van Gool, "Real-time facial feature detection using conditional regression forests," in Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, 2012, pp. 2578-2585: IEEE.

[6] X. Xiong and F. De la Torre, "Supervised descent method and its applications to face alignment," in Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), 2013, pp. 532-539.

[7] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," Journal of Machine Learning Research (JMLR), 9:1871–1874, 2008.

[8] P. Dollár, P. Welinder, and P. Perona, "Cascaded pose regression," in Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, 2010, pp. 1078-1085: IEEE.

[9] S. W. Chew, P. Lucey, S. Lucey, J. Saragih, J. F. Cohn, and S. Sridharan, "Person-independent facial expression detection using constrained local models," in Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on, 2011, pp. 915-920: IEEE.

[10] H. Gao, H. Ekenel, and R. Stiefelhagen, "Pose normalization for local appearance-based face recognition," Advances in biometrics, pp. 32-41, 2009.

[11] N. Wang, D. Tao, X. Gao, X. Li, and J. Li, "A comprehensive survey to face hallucination," International journal of computer vision (IJCV), vol. 106, no. 1, pp. 9-30, 2014.

[12] P. Viola and M. J. Jones, "Robust real-time face detection," International journal of computer vision (IJCV), vol. 57, no. 2, pp. 137-154, 2004.

[13] X. Cao, Y. Wei, F. Wen, and J. Sun, "Face alignment by explicit shape regression," International Journal of Computer Vision (IJCV), vol. 107, no. 2, pp. 177-190, 2014.

[14] L. Breiman, "Random forests," Machine learning, vol. 45, no. 1, pp. 5-32, 2001.

[15] V. Kazemi and J. Sullivan, "One millisecond face alignment with an ensemble of regression trees," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 1867-1874.

[16] S. Schulter, P. Wohlhart, C. Leistner, A. Saffari, P. M. Roth, and H. Bischof, "Alternating decision forests," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013, pp. 508-515.

[17] F. Moosmann, B. Triggs, and F. Jurie, "Fast discriminative visual codebooks using randomized clustering forests," in Conference on Neural Information Processing Systems (NIPS), 2006, vol. 2, p. 4.

[18] S. Schulter, C. Leistner, and H. Bischof, "Fast and accurate image upscaling with super-resolution forests," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3791-3799.

[19] J. Salvador and E. Pérez-Pellitero, "Naive bayes super-resolution forest," in Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015, pp. 325-333.

[20] P. N. Belhumeur, D. W. Jacobs, D. J. Kriegman, and N. Kumar, "Localizing parts of faces using a consensus of exemplars," transactions on pattern analysis and machine intelligence (TPAMI), vol. 35, no. 12, pp. 2930-2940, 2013.

[21] V. Le, J. Brandt, Z. Lin, L. Bourdev, and T. Huang, "Interactive facial feature localization," Computer Vision–ECCV 2012, pp. 679-692, 2012.

[22] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, 2005, vol. 1, pp. 886-893: IEEE.

[23] T. Ahonen, A. Hadid, and M. Pietikainen, "Face description with local binary patterns: Application to face recognition," transactions on pattern analysis and machine intelligence (TPAMI), vol. 28, no. 12, pp. 2037-2041, 2006.

[24] D. Cristinacce and T. F. Cootes, "Feature Detection and Tracking with Constrained Local Models," in BMVC, 2006, vol. 1, no. 2, p. 3.

[25] D. Cristinacce and T. Cootes, "Automatic feature localisation with constrained local models," Pattern Recognition, vol. 41, no. 10, pp. 3054-3067, 2008.

[26] I. Matthews and S. Baker, "Active appearance models revisited," International journal of computer vision (IJCV), vol. 60, no. 2, pp. 135-164, 2004.

[27] T. Baltrusaitis, P. Robinson, and L.-P. Morency, "Constrained local neural fields for robust facial landmark detection in the wild," in Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCV-Workshops), 2013, pp. 354-361.

[28] J. M. Saragih, S. Lucey, and J. F. Cohn, "Deformable model fitting by regularized landmark mean-shift," International Journal of Computer Vision (IJCV), vol. 91, no. 2, pp. 200-215, 2011.

[29] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework," International journal of computer vision (IJCV), vol. 56, no. 3, pp. 221-255, 2004.

[30] C.-J. Lin, R. C. Weng, and S. S. Keerthi, "Trust region newton method for logistic regression," Journal of Machine Learning Research (JMLR), vol. 9, no. Apr, pp. 627-650, 2008.

[31] A. Asthana, S. Zafeiriou, S. Cheng, and M. Pantic, "Robust discriminative response map fitting with constrained local models," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013, pp. 3444-3451.

[32] F. Zhou, J. Brandt, and Z. Lin, "Exemplar-based graph matching for robust facial landmark localization," in Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2013, pp. 1025-1032.

[33] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," International journal of computer vision (IJCV), vol. 104, no. 2, pp. 154-171, 2013.

[34] W. Liu et al., "SSD: Single shot multibox detector," in European Conference on Computer Vision (ECCV), 2016, pp. 21-37: Springer.

[35] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779-788.

[36] R. Weng, J. Lu, Y.-P. Tan, and J. Zhou, "Learning Cascaded Deep Auto-Encoder Networks for Face Alignment," IEEE Transactions on Multimedia (TMM), vol. 18, no. 10, pp. 2066-2078, 2016.

[37] J. Zhang, S. Shan, M. Kan, and X. Chen, "Coarse-to-fine auto-encoder networks (cfan) for real-time face alignment," in European Conference on Computer Vision (ECCV), 2014, pp. 1-16: Springer.

[38] D. C. Cireşan, U. Meier, L. M. Gambardella, and J. Schmidhuber, "Deep, big, simple neural nets for handwritten digit recognition," Neural computation, vol. 22, no. 12, pp. 3207-3220, 2010.

[39] A. Wang, J. Lu, J. Cai, T.-J. Cham, and G. Wang, "Large-margin multi-modal deep learning for RGB-D object recognition," IEEE Transactions on Multimedia (TMM), vol. 17, no. 11, pp. 1887-1898, 2015.

[40] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, "Facial landmark detection by deep multi-task learning," in European Conference on Computer Vision (ECCV), 2014, pp. 94-108: Springer.

[41] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in European Conference on Computer Vision (ECCV), 2014, pp. 184-199: Springer.

[42] J. Kim, J. Kwon Lee, and K. Mu Lee, "Accurate image super-resolution using very deep convolutional networks," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 1646-1654.

[43] C. Ledig et al., “Photo-realistic single image super-resolution using a generative adversarial network,” in IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (2017).

[44] N. Wang and D.-Y. Yeung, "Learning a deep compact image representation for visual tracking," in Advances in neural information processing systems (NIPS), 2013, pp. 809-817.

[45] Lucas, B. D. and Kanade, T. An iterative image registration technique with an application in stereo vision. In Seventh International Joint Conference on Artificial Intelligence (IJCAI-81), pages 674–679, Vancouver, 1981.

[46] S. Zhang, H. Yang, and Z. Yin, "Multiple deep convolutional neural networks averaging for face alignment," Journal of Electronic Imaging (JEI), vol. 24, no. 3, pp. 033013-033013, 2015.

[47] B. Sun, L. Li, G. Zhou, and J. He, "Facial expression recognition in the wild based on multimodal texture features," Journal of Electronic Imaging (JEI), vol. 25, no. 6, pp. 061407-061407, 2016.

[48] Y. Sun, B. Hu, J. Deng, and X. Li, "Supervised descent method with low rank and sparsity constraints for robust face alignment," in Sixth International Conference on Graphic and Image Processing (ICGIP 2014), 2015, pp. 944304-944304-6: International Society for Optics and Photonics.

[49] X. Ma, J. Liu, and W. Li, "Unified framework of face hallucination across multiple modalities," in Seventh International Conference on Machine Vision (ICMV 2014), 2015, pp. 94451T-94451T-5: International Society for Optics and Photonics.

[50] H. Zhou and K.-M. Lam, "Face hallucination using orthogonal canonical correlation analysis," Journal of Electronic Imaging (JEI), vol. 25, no. 3, pp. 033005-033005, 2016.

[51] H. Li, K.-M. Lam, M.-Y. Chiu, K. Wu, and Z. Lei, "Cascaded face alignment via intimacy definition feature," Journal of Electronic Imaging (JEI), vol. 26, no. 5, p. 053024, 2017.

[52] X. P. Burgos-Artizzu, P. Perona, and P. Dollar, "Robust face landmark estimation under occlusion," in IEEE International Conference on Computer Vision (ICCV), pp. 1513-1520, 2013

[53] X. Zhu and D. Ramanan, "Face detection, pose estimation, and landmark localization in the wild", in IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2012, pp. 2879-2886

[54] L. Liu, J. Hu, S. Zhang, and W. Deng, "Extended supervised descent method for robust face alignment," in Asian Conference on Computer Vision (ACCV), 2014 Workshops.

[55] F. Yu, S. Kumar, Y. Gong, and S.-F. Chang, "Circulant binary embedding," in International Conference on Machine Learning (ICML), 2014, pp. 946-954.

[56] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," Journal of machine learning research (JMLR), vol. 9, no. Aug, pp. 1871-1874, 2008.

[57] W.-S. Tam, C.-W. Kok, and W.-C. Siu, "Modified edge-directed interpolation for images," Journal of Electronic imaging (JEI), vol. 19, no. 1, pp. 013011, 2010.

[58] L. Zhang and X. Wu, "An edge-guided image interpolation algorithm via directional filtering and data fusion," IEEE Transactions on Image Processing (TIP), vol. 15, no. 8, pp. 2226-2238, 2006.

[59] X. Zhang and X. Wu, "Image interpolation by adaptive 2-D autoregressive modeling and soft-decision estimation," IEEE Transactions on Image Processing (TIP), vol. 17, no. 6, pp. 887-896, 2008.

[60] X. Li and M. T. Orchard, "New edge-directed interpolation," IEEE Transactions on Image Processing (TIP), vol. 10, no. 10, pp. 1521-1527, 2001.

[61] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, "Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval," IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), vol. 35, no. 12, pp. 2916-2929, 2013.

[62] R. Timofte, V. De Smet, and L. Van Gool, "Anchored neighborhood regression for fast example-based super-resolution," in Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2013, pp. 1920-1927.

[63] Y. Amit and D. Geman, "Shape quantization and recognition with randomized trees," Neural computation, vol. 9, no. 7, pp. 1545-1588, 1997.

[64] R. Zeyde, M. Elad, and M. Protter, "On single image scale-up using sparse-representations," in International conference on curves and surfaces, 2010, pp. 711-730: Springer.

[65] R. Timofte, V. De Smet, and L. Van Gool, "A+: Adjusted anchored neighborhood regression for fast super-resolution," in Asian Conference on Computer Vision (ACCV), 2014, pp. 111-126: Springer.

[66] P. H. Schönemann, "A generalized solution of the orthogonal Procrustes problem," Psychometrika, vol. 31, no. 1, pp. 1-10, 1966.

[67] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, 2010, pp. 3304-3311: IEEE.

[68] C. Cortes and V. Vapnik, "Support-vector networks," Machine learning, vol. 20, no. 3, pp. 273-297, 1995.

[69] D. Dai, R. Timofte, and L. Van Gool, "Jointly Optimized Regressors for Image Super-resolution," in Computer Graphics Forum, 2015, vol. 34, no. 2, pp. 95-104: Wiley Online Library.

[70] C. J. Burges, "A tutorial on support vector machines for pattern recognition," Data Mining and Knowledge Discovery (DMKD), vol. 2, no. 2, pp. 121-167, 1998.

[71] N. Cristianini and J. Shawe-Taylor, "An Introduction to Support Vector Machines and Other Kernel-based Learning Methods," Cambridge University Press, March 2000.

[72] D. Zhang and J. He, "Hybrid sparse-representation-based approach to image super-resolution reconstruction," Journal of Electronic Imaging (JEI), vol. 26, no. 2, pp. 023008, 2017.

[73] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in International Conference on Machine Learning (ICML), 1996, vol. 96, pp. 148-156.

[74] E. Agustsson, Timofte R., and L. Van Gool, "Regressor Basis Learning for Anchored Super-Resolution," in IEEE International Conference on Pattern Recognition (ICPR), Cancun, Mexico, Dec. 2016.

[75] W. Freeman, T. Jones, and E. Pasztor, "Example-based super-resolution, " IEEE Computer Graphics and Applications (CG&A), vol. 22, no. 2, 2002.

[76] A. N. Tikhonov, V. I. A. k. Arsenin, and F. John, "Solutions of ill-posed problems." Winston Washington, DC, 1977.

[77] J.-J. Huang, W.-C. Siu, and T.-R. Liu, "Fast image interpolation via random forests," IEEE Transactions on Image Processing (TIP), vol. 24, no. 10, pp. 3232-3245, 2015.

[78] R. Timofte, R. Rothe, and L. Van Gool, "Seven ways to improve example-based single image super resolution," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 1865-1873.

[79] J.-J. Huang and W.-C. Siu, "Learning Hierarchical Decision Trees for Single Image Super-Resolution," IEEE Transactions on Circuits and Systems for Video Technology (TCSVT), 2015.

[80] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in Proceedings of Eighth IEEE International Conference on Computer Vision (ICCV), 2001, vol. 2, pp. 416-423.

[81] Y. Zhang, Y. Zhang, J. Zhang, and Q. Dai, "CCR: Clustering and Collaborative Representation for Fast Single Image Super-Resolution," IEEE Transactions on Multimedia (TMM), vol. 18, no. 3, pp. 405-417, 2016.

[82] J. Jiang, X. Ma, C. Chen, T. Lu, Z. Wang, and J. Ma, "Single Image Super-Resolution via Locally Regularized Anchored Neighborhood Regression and Nonlocal Means," IEEE Transactions on Multimedia (TMM), vol. 19, no. 1, pp. 15-26, 2017.

[83] S. Schulter, P. Wohlhart, C. Leistner, A. Saffari, P. M. Roth, and H. Bischof, "Alternating decision forests," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013, pp. 508-515.

[84] S. Schulter, C. Leistner, P. Wohlhart, P. M. Roth, and H. Bischof, "Alternating regression forests for object detection and pose estimation," in Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2013, pp. 417-424.

[85] C. Dong, C. C. Loy, and X. Tang, "Accelerating the super-resolution convolutional neural network," in European Conference on Computer Vision (ECCV), 2016, pp. 391-407: Springer.

[86] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," Annals of statistics, pp. 1189-1232, 2001.

[87] J. Gall and V. Lempitsky, "Class-specific hough forests for object detection, " in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009.

[88] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," IEEE Transactions on Image Processing (TIP), vol. 19, no. 11, pp. 2861-2873, 2010.

[89] H. Chang, D.-Y. Yeung, and Y. Xiong, "Super-Resolution through neighbor embedding," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol. 1, pp. 275-282, 2004.

[90] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. Alberi-Morel, "Low-complexity single-image super-resolution based on nonnegative neighbor embedding," in the British Machine Vision Conference (BMVC), 2012.

[91] C.-Y. Yang and M.-H. Yang, "Fast direct super-resolution by simple functions," in Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 561-568, 2013.

[92] K. Zhang, D. Tao, X. Gao, X. Li, and Z. Xiong, "Learning multiple linear mappings for efficient single image super-resolution," IEEE Transactions on Image Processing (TIP), vol. 24, no. 3, pp. 846-861, 2015.

[93] Y. Zhang, Y. Zhang, J. Zhang, H. Wang, X. Wang, and Q. Dai, "Adaptive local nonparametric regression for fast single image super-resolution," in Visual Communications and Image Processing (VCIP), 2015, 2015, pp. 1-4: IEEE.

[94] J. Wang, J. Wang, J. Song, X.-S. Xu, H. T. Shen, and S. Li, "Optimized cartesian k-means," IEEE Transactions on Knowledge and Data Engineering (KDE), vol. 27, no. 1, pp. 180-192, 2015.

[95] Y. Zhang, K. Gu, Y. Zhang, J. Zhang, and Q. Dai, "Image super-resolution based on dictionary learning and anchored neighborhood regression with mutual incoherence," in IEEE International Conference on Image Processing (ICIP), 2015, Québec City, QC, Canada, Sep. 2015, pp. 591–595.

[96] B.-F. Wu, C.-C. Kao, C.-L. Jen, C.-R. Chiang, and P.-H. Lai, "Active Appearance Model Algorithm with K-Nearest Neighbor Classifier for Face Pose Estimation," Journal of Marine Science and Technology (JMST), vol. 22, no. 3, pp. 285-294, 2014.

[97] H. Li and K.-M. Lam, "Fast super-resolution based on weighted collaborative representation," in IEEE the 19th International Conference on Digital Signal Processing (DSP), pp. 914-918, 2014.

[98] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," IEEE transactions on pattern analysis and machine intelligence (TPAMI), vol. 38, no. 2, pp. 295-307, 2016.

[99] H. Li and K.-M. Lam, "Guided iterative back-projection scheme for single-image super-resolution," in IEEE Conference on Global High Tech Congress on Electronics (GHTCE), 2013, pp. 175-180.

[100] K. Lin, H.-F. Yang, J.-H. Hsiao, and C.-S. Chen. Deep learning of binary hash codes for fast image retrieval. In Computer Vision and Pattern Recognition Workshops (CVPRW), pages 27–35, 2015.

[101] H. Liu, R. Wang, S. Shan, and X. Chen, "Deep supervised hashing for fast image retrieval," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2064-2072.

[102] A. Gionis, P. Indyk, R. Motwani, et al. Similarity search in high dimensions via hashing. In VLDB, volume 99, pages 518–529, 1999.

[103] J. Wang, J. Yang, K. Yu, F. Lv, T. S. Huang, and Y. Gong. "Locality-constrained linear coding for image classification," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010.

[104] P. Li, X. Lu, and Q. Wang, "From dictionary of visual words to subspaces: locality-constrained affine subspace coding," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 2348-2357.

[105] I. Ram, M. Elad, and I. Cohen, "Image processing using smooth ordering of its patches," IEEE transactions on image processing (TIP), vol. 22, no. 7, pp. 2764-2774, 2013.

[106] A. Danielyan, V. Katkovnik, and K. Egiazarian, "BM3D frames and variational image deblurring," IEEE Transactions on Image Processing (TIP), vol. 21, no. 4, pp. 1715-1728, 2012.

[107] M. Irani and S. Peleg. "Improving resolution by image registration, "CVGIP: Graphical Models and Image Processing, 53(3), 1991

[108] H. Li, K.-M. Lam, and D. Li, "Joint Maximum Purity Forest with Application to Image Super-Resolution," Journal of Electronic imaging (JEI) 27(4), 043005 (2018), doi: 10.1117/1.JEI.27.4.043005.

[109] W. Dong, L. Zhang, G. Shi, and X. Wu, "Nonlocal back projection for adaptive image enlargement", IEEE International Conference on Image Processing (ICIP), pp. 349-352, November 2009.

[110] J. J. Rodriguez, L. I. Kuncheva, and C. J. Alonso, "Rotation forest: A new classifier ensemble method," IEEE transactions on pattern analysis and machine intelligence (TPAMI), vol. 28, no. 10, pp. 1619-1630, 2006.

[111] P. Kontschieder, M. Fiterau, A. Criminisi, and S. R. Bulo. "Deep neural decision forests," in Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 1467-1475, 2015.

[112] Zhou, Z. H., Feng, J. "Deep forest: Towards an alternative to deep neural networks", in International Joint Conference on Artificial Intelligence (IJCAI), 2017.

[113] Gelly, Sylvain, and David Silver. "Monte-Carlo tree search and rapid action value estimation in computer Go." Artificial Intelligence 175.11 (2011): 1856-1875.

[114] V. Mnih, K. Kavukcuoglu, D. Silver, A. Rusu, J. Veness, M. Bellemare, A. Graves, M. Riedmiller, A. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. Nature, 518(7540):529–533, 2015.

[115] A. Tamar, Y. Wu, G. Thomas, S. Levine, and P. Abbeel, "Value iteration networks," in Advances in Neural Information Processing Systems, 2016, pp. 2154-2162.

[116] D. Silver et al., "Mastering the game of Go with deep neural networks and tree search," Nature, vol. 529, no. 7587, pp. 484-489, 2016.

[117] Hosmer DW, Lemeshow S, Sturdivant RX. Applied Logistic Regression, 3rd Edition. New Jersey, USA: John Wiley & Sons; 2013.

[118] G. Valentini and F. Masulli, "Ensembles of learning machines," in Neural Nets WIRN Vietri-02, M. Marinaro and R. Tagliaferri, Eds. Heidelberg, Germany: Springer-Verlag, 2002, ser. Series Lecture Notes in Computer Sciences.

[119] Friedman J. Greedy boosting approximation: a gradient boosting machine. The Annals of Statistics 2001; 29(5): 1189-232.

[120] Freund, Y. and Shapire, R. E., "Experiments with a New Boosting Algorithm," In International Conference on Machine Learning. (ICML), 1996

[121] Dietterich TG. An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization. Machine Learning 2000; 40(2): 139-57.

[122] Schapire, R. E. (1990). The Strength of Weak Learnability. Machine Learning, 5(2):197{227. (page 18, 21, 22, 40)

[123] Schapire, R. E. and Freund, Y. (2012). Boosting: Foundations and Algorithms. MIT Press. (page 18, 19, 21)

[124] Freund, Y. and Mason, L., "The Alternating Decision Tree Learning Algorithm," In International Conference on Machine Learning. (ICML), 1999

[125] Rosenblatt, Frank. x. Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms. Spartan Books, Washington DC, 1961

[126] P. Viola and M. Jones, Robust Real-Time Face Detection, International Journal of Computer Vision, vol. 57, no. 2, pp. 137-154, 2004.

[127] Jordan, Michael I.; Bishop, Christopher M. (2004). "Neural Networks". In Allen B. Tucker. Computer Science Handbook, Second Edition (Section VII: Intelligent Systems). Boca Raton, Florida: Chapman & Hall/CRC Press LLC. ISBN 1-58488-360-X.

[128] Hastie, Trevor, Robert Tibshirani, Friedman, Jerome (2009). The Elements of Statistical Learning: Data mining, Inference, and Prediction. New York: Springer. pp. 485–586. ISBN 978-0-387-84857-0.

[129] MacQueen, J. B. (1967). Some Methods for classification and Analysis of Multivariate Observations. Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. 1. University of California Press. pp. 281–297.

[130] K. O. Bowman and L. R. Shenton, "Estimator: Method of Moments", pp 2092-2098, Encyclopedia of statistical sciences, Wiley (1998).

[131] J. Munkhammar, L. Mattsson, J. Rydén (2017) "Polynomial probability distribution estimation using the method of moments". PLoS ONE 12(4): e0174573. https://doi.org/10.1371/journal.pone.0174573.

[132] R. S. Sutton and A. G. Barto, Reinforcement learning: An introduction (no. 1). MIT press Cambridge, 1998.

[133] J. Friedman, T. Hastie, and R. Tibshirani, The elements of statistical learning. Springer series in statistics New York, 2001.

[134] D. M. Blei and P. Smyth, "Science and data science," Proceedings of the National Academy of Sciences, vol. 114, no. 33, pp. 8689-8692, 2017.

[135] D. P. Bertsekas, and D. P. Bertsekas, Dynamic programming and optimal control (no. 2). Athena scientific Belmont, MA, 1995.

[136] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural computation, vol. 9, no. 8, pp. 1735-1780, 1997.

[137] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," Neural computation, vol. 1, no. 2, pp. 270-280, 1989.

[138] R. Bellman. Dynamic Programming. Princeton University Press, 1957.

[139] V. Mnih, K. Kavukcuoglu, D. Silver, A. Rusu, J. Veness, M. Bellemare, A. Graves, M. Riedmiller, A. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. Nature, 518(7540):529–533, 2015.

[140] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In NIPS, 2012

[141] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in Advances in neural information processing systems (NIPS), 2014, pp. 3104-3112.

[142] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436-444, 2015.

[143] K. Narasimhan, T. Kulkarni, and R. Barzilay, "Language understanding for text-based games using deep reinforcement learning," arXiv preprint arXiv:1506.08941, 2015.

[144] C.-Y. Liou, W.-C. Cheng, J.-W. Liou, and D.-R. Liou. Autoencoder for words. Neurocomputing, 139:84–96, 2014.

[145] G. E. Hinton, "Deep belief networks," Scholarpedia, vol. 4, p. 5947, 2009.

[146] Dempster, A.P.; Laird, N.M.; Rubin, D.B. (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm". Journal of the Royal Statistical Society

[147] I. Dhillon and S. Sra. Generalized nonnegative matrix approximations with Bregman divergences. In Advances in Neural Information Processing Systems 17, Cambridge, MA, 2005. MIT Press.

[148] S. Banerjee and A. Roy, Linear Algebra and Matrix Analysis for Statistics. London: Chapman and Hall/CRC, 2014.

[149] Stone, James V. Independent component analysis: a tutorial introduction. Cambridge, MA, 2005. MIT Press.

[150] Jolliffe I.T. Principal Component Analysis, Series: Springer Series in Statistics, 2nd ed., Springer, NY, 2002, XXIX, 487 p. 28

[151] Permuter, H.; Francos, J.; Jermyn, I.H. (2003). Gaussian mixture models of texture and colour for image database retrieval. IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings (ICASSP '03)

[152] Yan, Xin, and Xiaogang Su. Linear regression analysis: theory and computing. World Scientific, 2009.

[153] Rennie, J., et al. "Tackling the poor assumptions of Naive Bayes classifiers," in International Conference on Machine Learning (ICML), 2003

[154] McLachlan, Geoffrey. Discriminant analysis and statistical pattern recognition. Vol. 544. John Wiley & Sons, 2004.

[155] Rokach, Lior, and Oded Z. Maimon. Data mining with decision trees: theory and applications. Vol. 69. World scientific, 2008.

[156] Altman, Naomi S. "An introduction to kernel and nearest-neighbor nonparametric regression." The American Statistician 46.3 (1992): 175-185.

[157] Z. Wang, D. Liu, J. Yang, W. Han, and T. Huang. "Deep networks for image super-resolution with sparse prior," in Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015.

[158] P. Indyk, R. Motwani, P. Raghavan, and S. Vempala, Locality-preserving hashing in multidimensional spaces, Proc. 29th ACM Symposium on Theory of Computing (STOC), pp. 618-625, 1997

[159] P. Indyk and R. Motwani. Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In 30th Symposium on Theory of Computing (STOC), 1998