Cascaded face alignment via intimacy definition feature

Hailiang Li,[a,b] Kin-Man Lam,[a] Man-Yau Chiu,[b] Kangheng Wu,[b] Zhibin Lei,[b]
[a]Department of Electronic and Information Engineering, The Hong Kong Polytechnic University
[b]Hong Kong Applied Science and Technology Research Institute Company Limited, Hong Kong
harley.li@connect.polyu.hk, enkmlam@polyu.edu.hk,
edmondchiu@astri.org, khwu@astri.org, lei@astri.org

Abstract. 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 paper, 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 the pose-indexed feature, more efficient than the histogram of oriented gradients (HOG) feature and the scale-invariant feature transform (SIFT) feature, and more compact than the local binary feature (LBF). Experimental validation of our algorithm shows that our approach achieves state-of-the-art performance when testing on some challenging datasets. Compared with the LBF-based algorithm, our method achieves about twice the speed, 20% improvement in terms of alignment accuracy and save an order of magnitude on memory requirement.

Keywords: cascaded face alignment; random forest; intimacy definition feature.

1. INTRODUCTION

Face alignment is a process to locate key-points and facial features (like eyebrows, eye corners, and mouth corners, see Fig. 1) from a given face image. It is an active research topic in computer vision. Face alignment 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 intense interest from the computer-vision research community.

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

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 model (CLM) [24, 25, 27, 28, 31] attempts to model shape prior to integration with local texture. It assumes that facial 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 the 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 achieves state-of-the-art performance, compared to 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 CLM 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 intensive computation for the inverse of the 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 numerous 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], face-landmark detection was improved through multi-task learning by designing a task-constrained deep model, with task-wise early stopping criterion to increase the learning convergence rate. In [37], deep neural network was exploited to learn feature-to-pose mapping functions by combining a cascaded framework for regressing pose-indexed features. To solve the inefficiency issue that appeared in the above-mentioned methods, an eight-learnable-layer deep convolutional neural network (DCNN), was proposed in [46], with rectified linear unit (ReLU) rather than the tanh activation function being used. This can achieve a speed five times faster in training convergence without decreasing its accuracy. To better initialize facial poses, in [36], a global exemplar-based deep auto-encoder network (GEDAN) was proposed 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 have achieved promising performance in terms of fitting accuracy, their heavy computation is a big obstacle to real-world applications, in particular, when hardware resources are limited or a graphics processing unit (GPU) is unavailable, such as mobile devices.

Therefore, a face-alignment algorithm, which is accurate, real-time, and small in size, is indispensable for the real-world industries, such as in the smart mobile phone applications**.** 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 features [1, 8, 13, 48], in which pose index provides some clue to 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 the pioneering works in [1, 5, 6, 8, 13, 48], in this paper, we propose a novel, discriminative and efficient feature, which can be incorporated into regression-based face-alignment frameworks to further boost their performance.

The remainder of this paper is organized as follows. In Section 2, we will review random-forest and random-forest-based cascaded face-alignment approaches. In Section 3, a feature derived from the pose-index feature, named intimacy definition feature (IDF), will be presented. Then, our proposed IDF-based cascaded random-forest face-alignment algorithm will be described and analyzed. Section 4 will evaluate our proposed method and compare it with recent fast local binary feature (LBF)-based methods. Section 5 will discuss how to cluster the training samples into subspaces for selecting representative shapes to form initial samples. Experiment results and parameter settings will be presented in Section 6, and conclusions and future work are given in Section 7.

2. Random Forests for Face Landmark Alignment

The landmark localization algorithm is important for face recognition and other related applications, which require extraction of local descriptors at some specified feature points or landmarks in a face. For face alignment, a number of points or landmarks, e.g., 17 or 68, are selected and searched from a face image. An example of the landmarks is shown in Fig. 1, in which 68 facial points are located around the eyes, nose, lips, and face contours. 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, there have been roughly three categories of face-alignment approaches followed by researchers. 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 learn a mapping from facial pixel appearance to shape increment [1, 5, 6, 8, 13].

The regression-based face-alignment approach tackles the face fitting problem by estimating mapping functions between the appearance and the shape-increment manifolds. Random forests are employed on regression-based algorithms in order to reduce the regressors' search complexity.

In our algorithm, we adopt the cascaded shape-regression paradigm that was first proposed by Dollar et al. [8] is an extension of 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.

2.1 Random Forests

Random forests [14] (RFs) have emerged recently as a very useful machine learning tool 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.

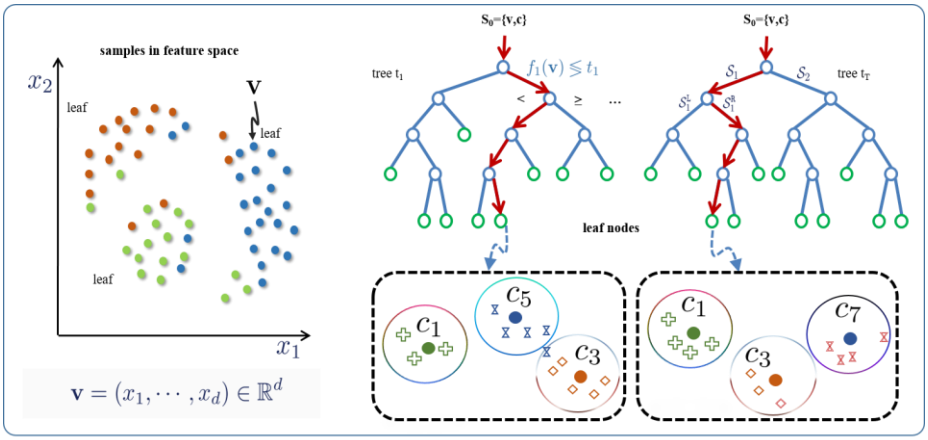


Fig. 2 An overview of random-forest-based clustering.

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

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:

$$y^* = \arg\max_y \frac{1}{\mathcal{T}} \sum_{t=1}^{\mathcal{T}} p_t(y|\boldsymbol{v}). \quad (1)$$

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

$$s(\boldsymbol{v}; \Theta) = \begin{cases} 0, & \text{if } \boldsymbol{v}(\Theta^i) < \Theta^t, \\ 1, & \text{otherwise,} \end{cases} \quad (2)$$

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:

$$I = \frac{|L|}{|L|+|R|} H(L) + \frac{|R|}{|L|+|R|} H(R), \quad (3)$$

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 their 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 (4), but it can be replaced by variance [1] or the Gini index [14].

$$H(S) = -\sum_{k=1}^{K} [p(k|S) * \log(p(k|S))] \quad (4)$$

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.

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:

$$S_t = S_{t-1} + R_t\big(\phi_t(I, S_{t-1})\big), \tag{5}$$

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 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 a similar workflow, as shown in Fig. 5.

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) between two members (leaf-nodes) in a binary family tree.

3.1 Efficient Metric on Intimacy Definition Feature

To explain the features, we use a family member structure to illustrate the binary tree in the random-forests scheme, as shown in Fig 3. In this structure, each leaf-node represents a family

member, and the relationship between two members is measured by their DoI values, which can be computed by their respective intimacy definition feature (IDF) values. In Fig. 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.

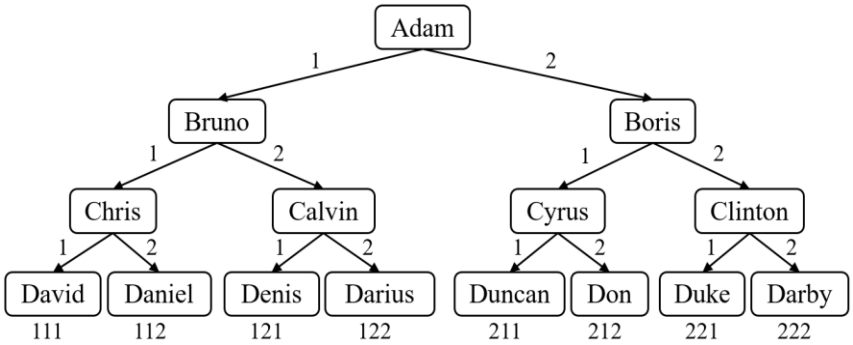


Fig. 3 A family tree with the degree of intimacy (DoI) values of family members in the 4th generation.

As we can see in the family tree in Fig. 3, two persons, who share more adjacent 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 paper, 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:

$$IDF = \sum_{l=1}^{L} path\_value_l * k^l, \quad (6)$$

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 any two members in a family tree by their IDT values. Through the family tree, as constructed in Fig. 3, the family members (nodes) can be replaced by visual features, which are then encoded by IDF values. Consequently, the similarity between two family members (nodes) can be measured by computing their DoI values.

In our study, we found that this simple, yet efficient, feature computed by traveling a tree in a random forest can achieve promising performance, in terms of both accuracy and speed, as shown in Section 4. 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:

$$nomalized\_IDF = \frac{(IDF - IDF_{min})}{(IDF_{max} - IDF_{min})}, \quad (7)$$

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. Then, the normalized IDF value for David (**111**) can be calculated as: (111−100)/(222−100) = **0.090164**.

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.

As with the LBF [1] feature, this locally learned pose-indexed features is not used, because it is not sufficiently discriminative, and does not explicitly encode the path of a sample along a tree. 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, each IDF value, encoded in a leaf-node, is a floating-point number, which can achieve high dimensionality reduction, compared 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 features, which can 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 our pose-indexed feature is described in Fig. 4.

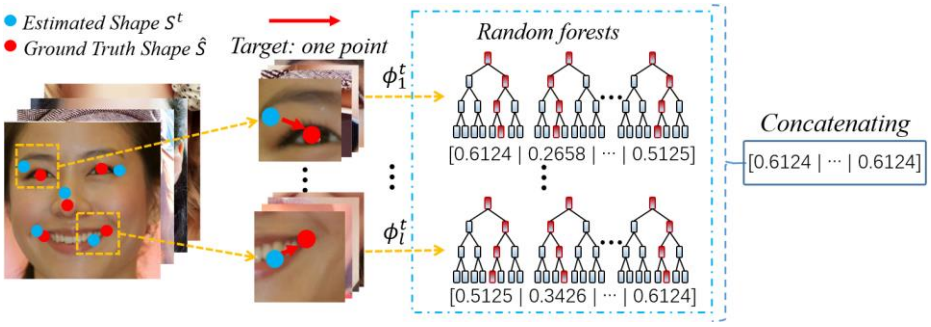


Fig. 4 The process of IDF-based feature vector extraction

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

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, 51] used in linear regression. The insight of SDM is to

directly learn shape increments from appearance changes, which can be viewed to estimate the conditional likelihood function $p(y|x)$, where y and x are the shape increment and appearance, respectively. Meanwhile, SDM employs a complicated non-linear operator for feature extraction (e.g. the HOG feature or SIFT feature [6, 51]), which slows down its speed when handling more faces in some challenging scenarios. From a theoretical perspective, SDM can be regarded as an extension of the Lucas & Kanade (LK) algorithm [45]. The LK algorithm, which holds an assumption that a linear relationship can be estimated from pixel appearance to geometric displacement, is worked as a classic optical flow algorithm, for tackling image and object-alignment problems.

In [1], random forests were used for training, by minimizing 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). In our proposed algorithm, IDF replaces LBF in the cascaded alignment framework, as depicted in Fig. 5. The success of LBF in [1] is due to its feature-learning step, where features are explicitly learned for the given specific task. Due to the sparse nature of the LBF feature vector, the inference phase can be reduced to traversing the forest, and performing simple table look-ups and additions. The authors in [1] claimed that the LBF method can achieve an impressive speed of approximately 3,000 fps, (with tailored setting on some parameters), in its fast version.

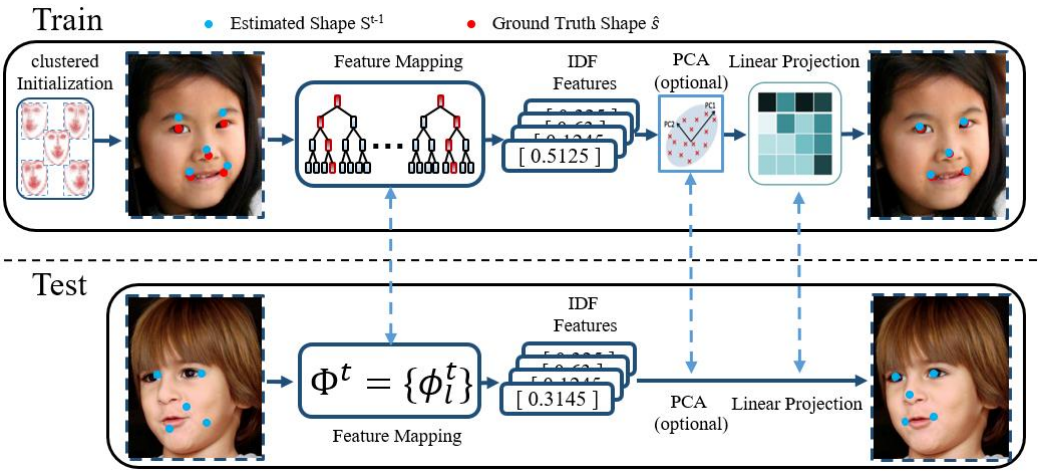


Fig. 5 An overview of the workflow for IDF-based cascaded regression face alignment.

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. However, LBF is a binary, sparse feature, and carries labelling information, which makes PCA not applicable. To avoid the computational complexity, the LBF-based approach should limit the tree depth to a relatively small value, e.g. 5, which means that there are, at most, 16 leaf-nodes in each tree. Consequently, this heavily restricts its capability for classification and regression.

Compared to the pixel-based 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 can also encode the path of a sample along a tree. However, the performance is very poor, because index values are similar to labels, which make the results inclined to be over-fitting. A simple analysis in Fig. 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. 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, 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 paper is that the efficient IDF feature is proposed to replace the LBF feature. This can greatly reduce the feature dimensionality, while a promising performance can still be achieved. Therefore, our algorithm runs much faster and requires less memory than that using LBF. For example, for the setting: $l = 68$, $t = 10$, and $d = 7$, the feature dimensionality of IDF is 68×10×1 = 680, rather than 43,520 (=68×10×64) for LBF. In other words, the dimensionality is reduced by 64 times.

4. Validation Results and Comparison to the LBF Feature

To validate the effectiveness, efficiency, and less memory requirement of our proposed IDF-based face-alignment method, we conducted intensity experiments on some public datasets, and compared the performances of our method with LBF [1].

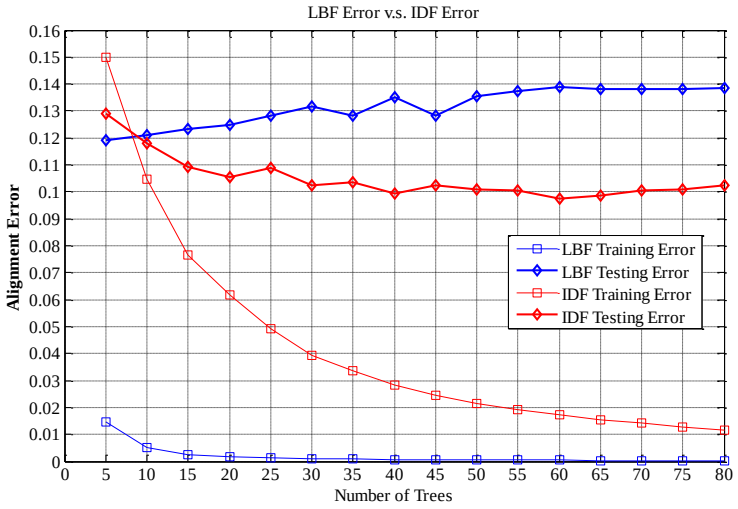


Fig. 6 A comparison of the alignment errors of the IDF vs LBF algorithms on the LFPW dataset [20], with tree depth = 7, number of training samples = 500, and number of testing samples = 300.

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 the LBF and the IDF feature. Fig. 6 shows the alignment errors in the training and testing stages, based on the LFPW dataset [20], with different numbers of trees. 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.

Table-1 Alignment errors at different stages, with different number of trees, based on the LBF algorithm. (LFPW dataset [20], number of training samples = 300, number of testing samples = 100)

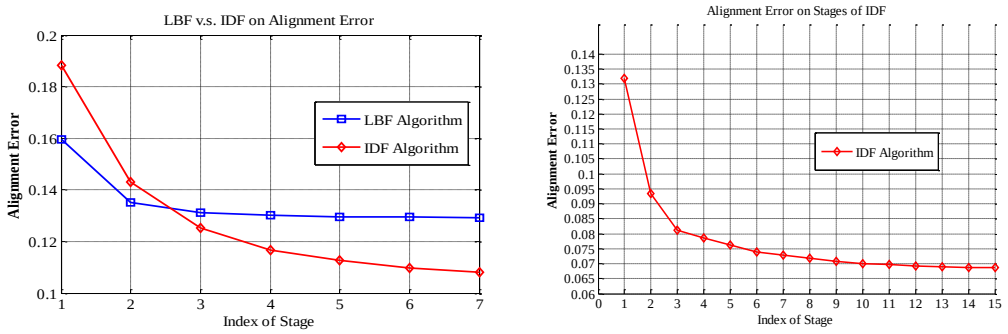
	Number of Trees									
Stage	5	10	20	30	40	50	60	70	80	*Avg.*
1	0.1765	0.1714	0.1630	0.1583	0.1583	0.1583	0.1533	0.1495	0.1485	**0.1597**
2	0.1411	0.1341	0.1410	0.1315	0.1326	0.1315	0.1397	0.1387	0.1390	**0.1352**
3	0.1386	0.1382	0.1390	0.1295	0.1293	0.1292	0.1252	0.1251	0.1276	**0.1312**
4	0.1385	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-2 Alignment errors at different stages, with different number of trees, based on the IDF algorithm. (LFPW dataset [20], number of training samples = 300, number of testing samples = 100)

	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**

Another factor we should consider is the number of trees required to achieve a specific alignment error. From Fig. 6, we can see that using about 10 trees in 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 at a steeper rate than LBF. In other words, IDF converges faster in the coarse-to-fine search, because it has a higher discriminative power than LBF.

Fig. 7(a) illustrates the alignment errors of the LBF and IDF methods, with different numbers of stages (with 300 samples for training and 100 samples for testing). We can see that the curve for IDF is much steeper than that for LBF, which means that the IDF feature is more discriminative than LBF and achieves a higher convergence rate at later stages. An explanation for this is that the IDF value is represented as a floating-point number, which has a stronger representation than a LBF binary value. Fig. 7(b) shows the alignment errors of IDF, with more stages. We can see that the alignment error reduces when the number of stages increases. To obtain a balance between computational complexity and fitting accuracy, using 7 stages is a compromise. Therefore, in the rest of this paper, our algorithm uses 7 stages in all experiments.



(a) (b)

Fig. 7 Alignment errors with different numbers of stages: (a) LBF vs IDF, up to 7 stages, and (b) IDF only, up to 15 stages (tree depth = 7, LFPW dataset [20], number of training sample = 300, number of testing samples =100).

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, requires the same order of 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, when compared to the LBF-based algorithms. As shown in Fig. 8, the number of frames processed per second, based on IDF, is about 2 times faster than LBF, with the same setting.

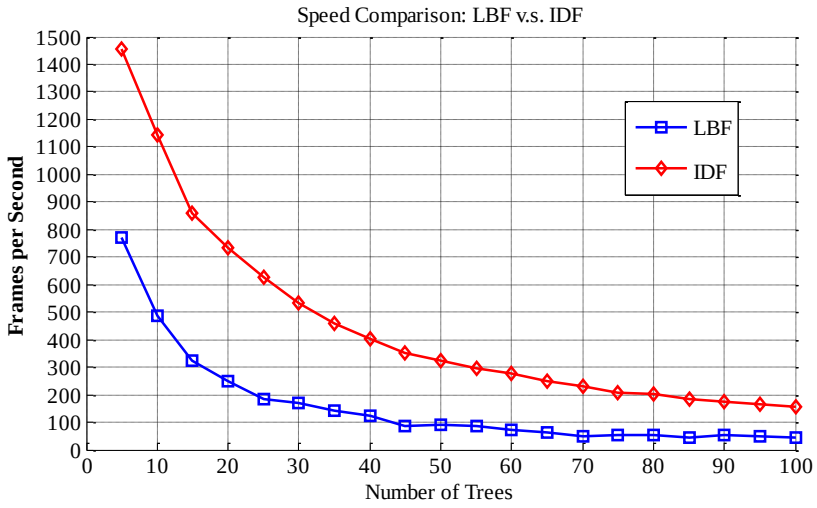


Fig. 8 The speed in terms of number of frames per second for the IDF vs LBF algorithms (tree depth = 7, Helen dataset [21]).

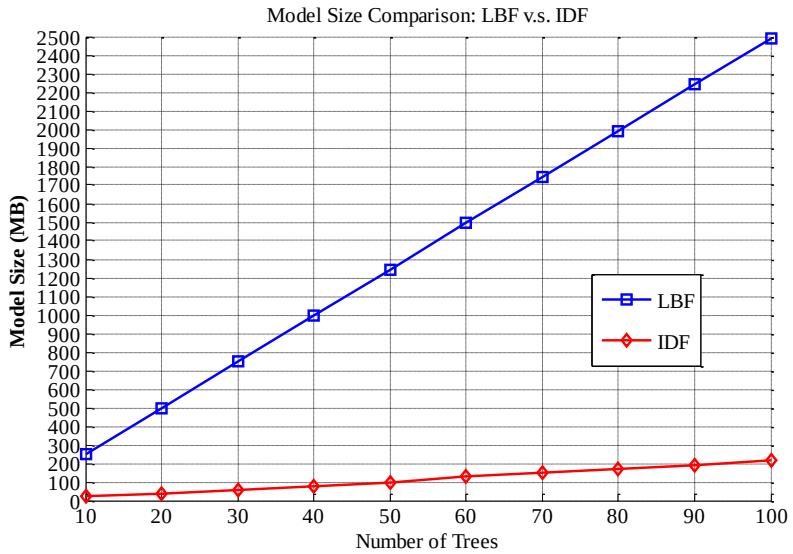


Fig. 9 Memory requirements (MB) of IDF vs LBF with different numbers of trees (tree depth: 7, Helen dataset [21]) at the inference stage.

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

5. Training with Initial Shapes from Similar Samples Spanned Subspaces

Sensitivity to the initial shape is a limitation of regression-based models. This means that using a mean face as the initial shape will likely result in unsatisfactory performance on 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, when 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, there is an overlooked issue that the tedium of labeling pose faces manually will cause mistakes in the labeling results, as well as being imprecise. For example, it is ambiguous whether human eyes can discriminate between a face with a pose with a 45-degree angle from another face with a 30-degree angle or a 60-degree angle.

In [2], a pose detector, 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, is employed for estimating initial shapes, based on the k nearest neighbors selected from training samples. 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, a 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 requires an additional stage for shape initialization.

To further improve the performance, we refine the face initialization by using the k-means clustering algorithm. Different from the above-mentioned two methods [2, 5], our algorithm does not use any pose detector or search for similar faces from a large database. 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 our algorithm, we propose 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 pose, expression, illumination, and with partial occlusions) and the Helen dataset [21] (2000 training + 330 testing images, which exhibit a large variation in appearance, such as pose, expression, ethnicity, age and gender, as well as the general imaging and environmental conditions).

Fig. 10 Clustering 7 groups of face images with different poses through the k-means clustering algorithm.

We use the k-means algorithm to cluster the training samples into a number of groups, as shown in Fig. 10. Then, for each target face image, instead of using 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 the training stage. Therefore, the model is learned with the pose information from the spanned

pose space of selected neighboring examples, which can represent the target faces well. Experimental results in Fig. 11 show that the "IDF + Clustering" training scheme can further improve the alignment error, when compared to the non-clustering scheme.

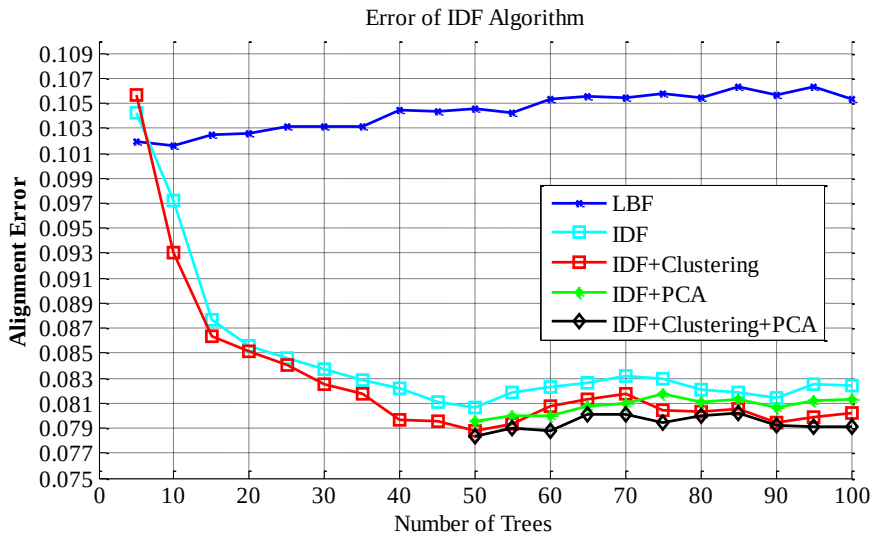


Fig. 11 Alignment errors of the IDF algorithm, with and without using clustering and PCA (tree depth: 7, stages: 5, Helen dataset [21]).

Table-3 Feature dimensions of IDF, with and without 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 larger the number of linear-regression weights is required for the regression model. This results in more computations and memory in the inference stage, because all the weights of the models for the cascaded stages are required to be loaded into memory. Another advantage of using IDF is that, compared to LBF, it can apply PCA to reduce its feature dimensionality, because IDF is represented by floating-point numbers. From Table 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 relaxation on linear regression after dimension reduction, theoretically, an optimal and faster solution can be found when the feature dimension of IDF increases. However, it is hard to apply PCA to the LBF binary boolean-like values. Therefore, IDF with a higher dimensionality can be adopted to achieve both efficiency and accuracy, which is impossible for the LBF feature. Fig. 5 shows the whole workflow of the proposed algorithm, and the training and fitting stages are described in **Algorithm** 1 and **Algorithm** 2, respectively.

Algorithm 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, $\bar{S}^i$ is the initial shape for S^i, 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 $S = \{S^i\}$ 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 the 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 the shape increment: ΔS_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 2 IDF Fitting Stage:

Input: Testing face image I, the initial (mean) shape S_0 obtained from training samples, trained regressors: $R = (R_1, \ldots, R_T)$, T: stage count.

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: $\phi_t(I, S_{t-1})$

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

6. Experimental Results and Parameter Settings

By analyzing the encoding process of IDF, it is found that the IDF value of each node in a random forest 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.

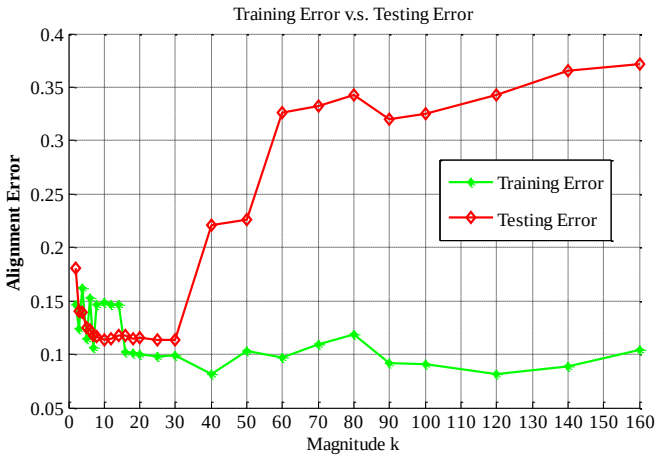


Fig. 12 Alignment errors for different magnitude values of k.

As shown in Fig. 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.

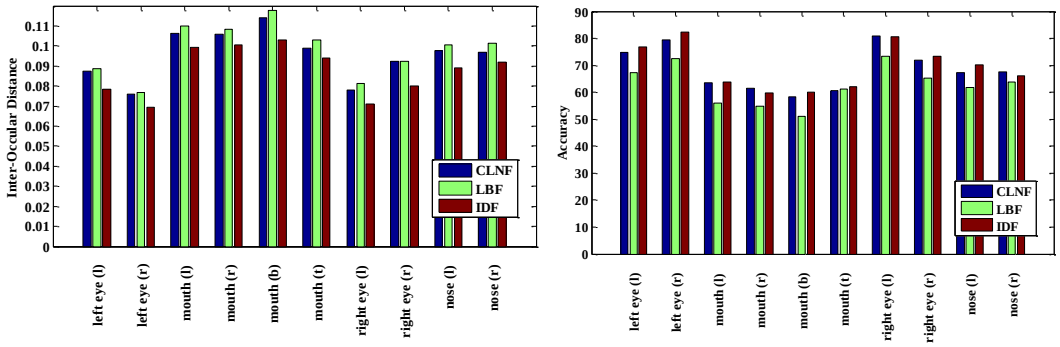


Fig. 13 Comparison of LBF [1], CLNF[27] and IDF, with performance on accuracy and InterOccular distance criterion on 10 facial landmark points in the Helen dataset.

The performance of the IDF method, LBF [1], and CLNF [27], in terms of accuracy and the inter-ocular distance criterion, for different facial landmarks (with 10 facial landmark points) are shown in Fig. 13. The results demonstrate that our proposed IDF-based method is comparable to or, in many cases, outperforms recent state-of-the-art methods. Fig. 13 also shows that, based on these two criteria, locating the facial landmarks around the mouth region is the most challenging for all the methods. This is because the landmarks around the mouth region suffer from significant variations caused by facial-expression changes. For the landmarks in the mouth region, our proposed IDF-based method achieves better performances than the same regression-based method with LBF and is comparable to the classic CLM-based method, and the CLNF [27] method.

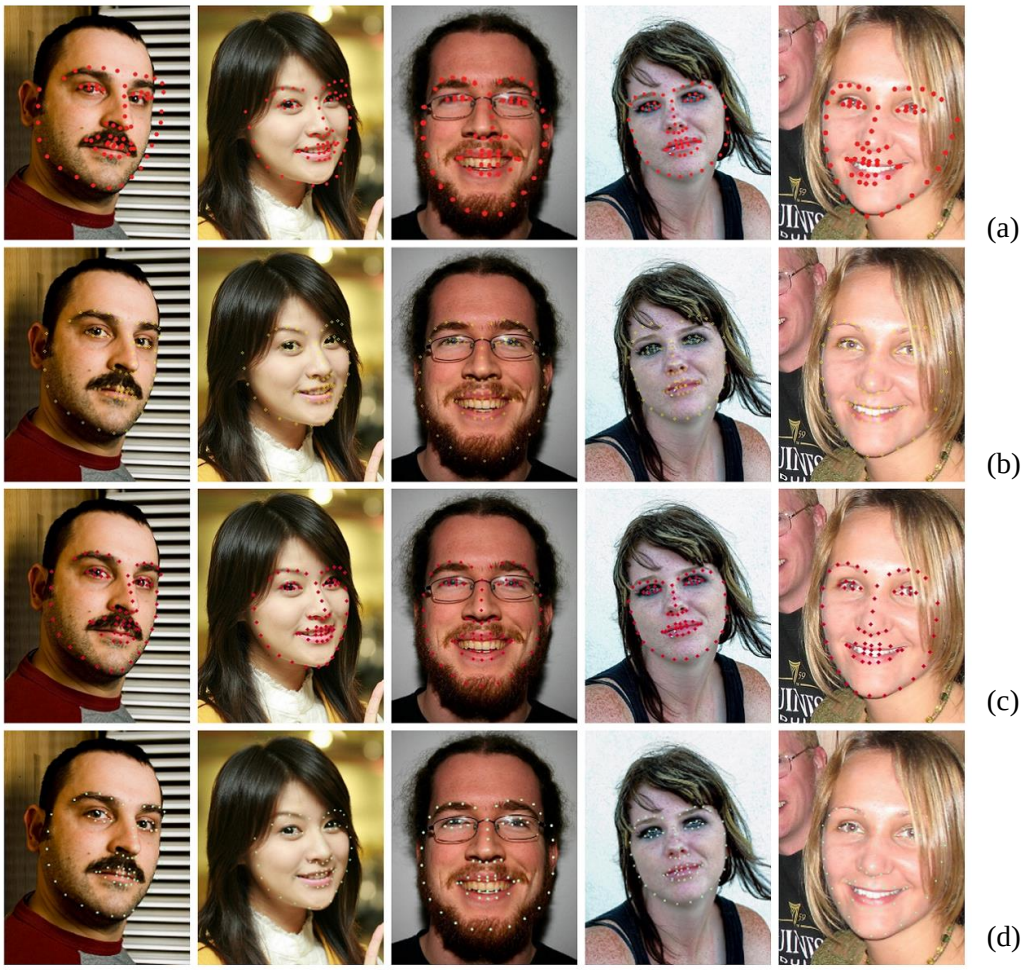


Fig. 14 Fitting results of different methods, with 68 landmarks, on the Helen dataset: (a) LBF [1], (b) One-Milli-Second [15], (c) CLNF[27], and (d) IDF.

Fig. 14 demonstrates some visual 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] was used for both LBF and IDF, and the linear regression type was set at L2R_L2LOSS_SVR, i.e. L^2-regularized L^2-loss support vector regression (primal), in which the Newton method with trust-region step control is employed to achieve faster convergence [30].

7. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a novel, simple, but effective, and discriminative feature, and explored the random-forest-based cascaded regression model for face alignment. The proposed feature, intimacy definition feature (IDF), is constructed with a full binary family tree by computing the degree of intimacy (DoI) of any two members in the same tree level. The DoI values can encode the path from the root to a leaf-node with a floating-point value.

The contributions of the paper 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, 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, which form a subspace. With this initialization strategy, the cascaded regression approach is capable of learning a 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 will enrich research based on random forests.

Presently, 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 features, e.g. the SIFT feature, so further investigation is necessary to tackle these problems for faces with noise, large poses and occlusion.

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, 2014, pp. 1685-1692.
2. H. Zhou, K.-M. Lam, and X. He, "Shape-appearance-correlated active appearance model," Pattern Recognition, 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, 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, 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, vol. 9, no. Aug, pp. 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, 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, 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, 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, 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, 2013, pp. 508-515.
17. F. Moosmann, B. Triggs, and F. Jurie, "Fast discriminative visual codebooks using randomized clustering forests," in 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, 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, 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, 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, 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, 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, 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, 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, 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, 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, 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, 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, vol. 104, no. 2, pp. 154-171, 2013.
34. W. Liu et al., "SSD: Single shot multibox detector," in European Conference on Computer Vision, 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, 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, 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, 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, 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, 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, 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 Conference on Computer Vision and Pattern Recognition, 2016, pp. 1646-1654.
43. C. Ledig et al., "Photo-realistic single image super-resolution using a generative adversarial network," arXiv preprint arXiv:1609.04802, 2016.
44. N. Wang and D.-Y. Yeung, "Learning a deep compact image representation for visual tracking," in Advances in neural information processing systems, 2013, pp. 809-817.
45. B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," 1981.
46. S. Zhang, H. Yang, and Z. Yin, "Multiple deep convolutional neural networks averaging for face alignment," Journal of Electronic Imaging, 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, 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, vol. 25, no. 3, pp. 033005-033005, 2016.
51. X. Xiong and F. De la Torre, "Global supervised descent method," in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, June 2015, pp. 2664–2673.

Hailiang Li received his M.Sc. degree from department of automation of Xiamen University, China, in 2004. He is studying in the Hong Kong Polytechnic University as a part-time PhD student. His research interests include image super-resolution, face alignment and Bayesian Inference. Currently, He works as a software engineer with Hong Kong Applied Science and Technology Research Institute (ASTRI) and his work is related to image processing, computer vision and machine learning.

Kin-Man Lam is a professor of the Department of Electronic and Information Engineering, the Hong Kong Polytechnic University. He is the VP-Member Relations and Development of the Asia-Pacific Signal and Information Processing Association (APSIPA) and the Director-Membership Services of the IEEE Signal Processing Society. He serves as an Associate Editor of Digital Signal Processing, APSIPA Trans. on Signal and Information Processing. His research interests include human face recognition, image and video processing, and computer vision.

Man-Yau Chiu received engineering doctorate degree in Electronic and Information Engineering from the Hong Kong Polytechnic University in 2011. He is now working in Hong Kong Applied Science Technology Research (ASTRI) as a senior engineer and works on various research and development projects related to latest video technology, computer vision and machine learning. His research interests include fast motion estimation, scalable video coding, computer vision and machine learning.

Kangheng Wu is the Principle Engineer of Hong Kong Applied Science and Technology Research Institute (ASTRI). He received his BS, MS, and PhD degree from Sun Yat-Sen University, China, in 2000, 2003 and, 2006 respectively. He has over 10 years of experience in cloud computing, machine learning, algorithm trading, robo-advisor, block-chain, smart water, video analysis and P2P networking. He has multiple patents and published papers on various journals, IEEE and ACM conference proceedings.

Zhibin Lei is the acting director of intelligent software and systems of ASTRI. He has worked at Bell Labs, Lucent Technologies, Panasonic Research, and the Chinese University of Hong Kong. He has more than 100 publications in journals, conferences, and patent applications. He has got a meritorious award for international mathematical contest in modeling by SIAM in 1989. He obtained Bachelor′s degree from Beijing University, and PhD degree from Brown University.