Face Alignment via Regressing Local Binary Features

Shaoqing Ren, Xudong Cao, Yichen Wei, and Jian Sun

***Abstract*— This paper presents a highly efficient and accurate regression approach for face alignment. Our approach has two novel components: 1) a set of local binary features and 2) a locality principle for learning those features. The locality principle guides us to learn a set of highly discriminative local binary features for each facial landmark independently. The obtained local binary features are used to jointly learn a linear regression for the final output. This approach achieves the state-of-the-art results when tested on the most challenging benchmarks to date. Furthermore, because extracting and regressing local binary features are computationally very cheap, our system is much faster than previous methods. It achieves over 3000 frames per second (FPS) on a desktop or 300 FPS on a mobile phone for locating a few dozens of landmarks. We also study a key issue that is important but has received little attention in the previous research, which is the face detector used to initialize alignment. We investigate several face detectors and perform quantitative evaluation on how they affect alignment accuracy. We find that an alignment friendly detector can further greatly boost the accuracy of our alignment method, reducing the error up to 16% relatively. To facilitate practical usage of face detection/alignment methods, we also propose a convenient metric to measure how good a detector is for alignment initialization.**

***Index Terms*— Face alignment, tracking, random forest, local binary feature.**

I. Introduction

DISCRIMINATIVE shape regression has emerged as a leading approach for accurate and robust face alignment [1]–[15]. This is primarily because these approaches have some distinct characteristics: 1) they are purely discriminative; 2) they are able to enforce shape constrains adaptively; 3) they are capable of effectively leveraging large bodies of training data.

The shape regression approach predicts facial shape S in a cascaded manner [1], [3], [5]–[7], [9]–[11]. Beginning with an initial shape S^0, S is progressively refined by estimating a shape increment ΔS stage-by-stage. In a generic form, a shape increment ΔS^t at stage t is regressed as:

$$\Delta S^t = W^t \Phi^t \left(I, S^{t-1} \right), \tag{1}$$

where I is the input image, S^{t-1} is the shape from the previous stage, Φ^t is a feature mapping function, and W^t is a linear regression matrix. Note that Φ^t depends on both I and S^{t-1}. A feature learned in this way is referred to as a "shape-indexed" feature [1], [7]. The regression goes to the next stage by adding ΔS^t to S^{t-1}.

The feature mapping function Φ^t is essential in shape regression. In previous works, it is either designed by hand [6], [10], [11], [16] or by learning [1], [7], [9]. The process in [6] simply uses SIFT features for feature mapping and trains W^t by a linear regression. While this simple approach works well, the handcrafted general purpose features are not optimal for specific face alignment. In contrast, the processes in [1], [7], and [9] learn both Φ^t and W^t jointly by a tree based regression on the whole face region in a data-driven manner.

In principle, the latter learning based approach should be better because it learns task-specific features. However, as reported in existing literature, it is only on par with the approach using a hand-designed SIFT feature. It is inferred that this is due to two issues caused by the overly high freedom of Φ^t. The first is a practical issue. Using the entire face region as the training input results in an extremely large feature pool, which translates into unaffordable training costs if we want to learn the most discriminative feature combination. The second is a generalization issue, which is more crucial. The large feature pool has many noisy features. This can easily lead to over fitting and damage performance in testing.

In this work, we propose a better learning based approach. It regularizes learning with a "locality" principle. This principle is based on two insights: for locating a certain landmark at a stage, 1) the most discriminative texture information lies in a local region around the estimated landmark from the previous stage; 2) the *shape context* (locations of other landmarks) and *local texture* of this landmark provide sufficient information. These insights imply that we may first learn intrinsic features to encode the local texture for each landmark independently, then perform joint regression to incorporate the shape context.

We propose the following two types of regularization for learning Φ^t:

- Φ^t is decomposed into a set of independent local feature mapping functions, i.e. $\Phi^t = [\phi_1^t, \phi_2^t, \ldots, \phi_L^t]$ (L is the number of landmarks).

Manuscript received July 9, 2015; revised November 24, 2015; accepted January 12, 2016. Date of publication January 8, 2016; date of current version January 29, 2016. This work was supported by the National Natural Science Foundation of China under Grant 61331015 and Grant 61473271. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Wen Gao.

S. Ren is with the University of Science and Technology of China, Hefei 230026, China (e-mail: sqren@mail.ustc.edu.cn).

X. Cao was with Microsoft Research, Beijing 100080, China. He is now with SenseTime (e-mail: notcxd@gmail.com).

Y. Wei and J. Sun are with the Visual Computing Group, Microsoft Research, Beijing 100080, China (e-mail: yichenw@microsoft.com; jiansun@microsoft.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2016.2518867

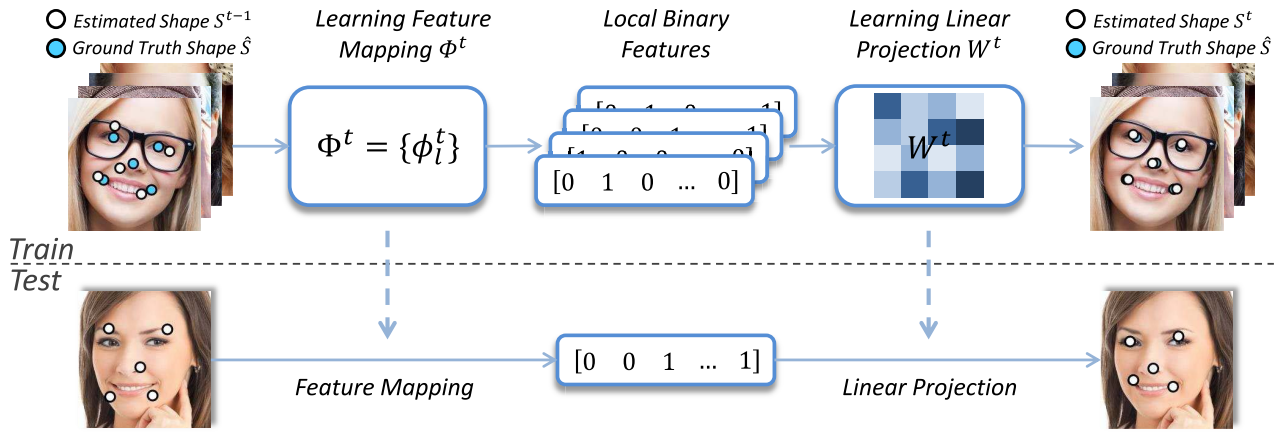


Fig. 1. Overview of our approach. In the training phase, we begin by learning a feature mapping function $\Phi^t(I_i, S_i^{t-1})$ to generate local binary features. Given the features and target shape increments $\{\Delta S_i^t = \hat{S}_i - S_i^{t-1}\}$, we learn a linear projection W^t by linear regression. In the testing phase, the shape increment is directly predicted and applied to update the current estimated shape.

- Each ϕ_l^t is learned by independently regressing lth landmark, in the corresponding *local* region.

The proposed regularization can effectively screen out the majority of noisy or less discriminative features, reduce learning complexity, and lead to better generalization.

To learn each ϕ_l^t, we use an ensemble-trees based regression to *induce* binary features. The binary features encode the intrinsic structure in a local region for predicating the landmark position. After concatenating all *local binary features* to form the feature mapping Φ^t, we discriminatively learn W^t for global shape estimation. We find that our two-step learning process (local binary features and global linear regression) is much better than the one-step joint learning of Φ^t and W^t by tree based regression in [1] and [7].

In addition to better accuracy, our approach is also much more efficient. Because the local binary features are tree based and highly sparse, the process of extracting and regressing such features is extremely rapid. We show that a fast version of our approach runs at 3,000+ frames per second (FPS) on a desktop computer with a single CPU thread[1] and achieves comparable results with state-of-the-art methods. Our accurate version runs at 300+ FPS and significantly outperforms state-of-the-art equivalents in terms of accuracy on a variety of benchmarks. The high speed of our approach is crucial for scenarios and devices where computational power is limited and computational budget is a major concern. For example, our fast version still runs at 300 FPS on a modern mobile phone with single cpu thread.[2] To the best of our knowledge, this is the first approach that achieves such high FPS on mobile phone using a single CPU core. This opens up new opportunities for all online face applications.

Besides the alignment algorithm, the face detector used to initialize alignment also greatly affects the performance of alignment, however, this key issue has received less attention in previous literature. Currently, most face alignment methods start with the output rectangles of a face detector. The difficulty of the learning problem of alignment is high correlated to the quality of the face detector. In real applications, a face rectangle with poor quality indicates a high chance of alignment failure. If the detector can be specifically designed or optimized for alignment, there should be a considerable performance gain for alignment. While it has been widely accepted that face alignment can help detection [17]–[19], few works study how much an alignment-friendly detector can improve alignment performance.

In this paper, we show that the potential improvement from an alignment-friendly detector can be obtained. Specifically, an alignment-friendly detector [19] based on the proposed alignment method can achieve more than 16% relative error reduction compared with the official face rectangles from 300-W [20]. This inspires us to design an alignment-friendly detector and jointly optimize detection and alignment.

Compared with related conference paper, in this journal version, we make more comparisons and take more discussions for the proposed alignment method. In addition, we study the impact of initialization for face alignment and discusses how to obtain better initialization. As an additional product of these explorations, we come up with a stronger version of LBF with 'box refine'.

II. Related Works

Active Appearance Models (AAM) [21] solve the face alignment problem by jointly modeling holistic appearance and shape. Many improvements over AAM have been proposed [22]–[27]. Instead of modeling holistic appearances, the "Constrained Local Model" [4], [18], [28]–[33] learn a set of local experts (detectors [29], [31], [32], [34], [35] or regressors [2], [4], [30]) and constrains them using various shape models. These approaches are better for generalization and robustness.

Our work belongs to the shape regression approach category. As one of the most popular frameworks for face alignment, many works [1]–[7], [16] share the generic form of this category.

In this framework (Equation 1), regression matrix W^t and feature mapping function Φ^t are the most important parts, which distinguish between different approaches. Both Cao et al. [1] and Burgos-Artizzu et al. [7] use boosted ferns

[1]Intel i7-2600 CPU @3.4GHz, 8GB RAM
[2]Snapdragon MSM8260A CPU @1.5GHz, 1GB RAM

(a variant of tree) to learn Φ^t. With the learnt tree structure, W^t is locally counted within each tree node. These methods benefit from the highly efficient inference of ensemble trees, with the potential to be super fast. However, the large amount of trees used restricts speed. Our novel learning method greatly enhances the efficiency of the ensemble trees, achieving better performance with fewer trees. Besides tree based learnt features, hand-crafted features (e.g, SIFT [6] and HOG [16]) are also popular. Xiong and De la Torre [6] use concatenated SIFT features of each landmark as the feature and computes the regression matrix via linear regression. Global regression and strong features are two key parts of this method. While the heavy computational cost of the strong SIFT feature is a burden for computational sensitive situations.

Recently, deep learning based alignment methods are popular and have shown high accuracy [8], [11], [12]. Sun et al. [8] explores cascade convolution neural network (CNN) for face alignment. Zhang et al. [11] proposes a coarse-to-fine auto-encoder networks for high accuracy face alignment. Zhang et al. [12] improves CNN based face alignment by face related multi-task learning. These deep learning based methods are very attractive for their strong performance, while due to the high computational cost of neural network, these methods are usually 1-2 orders of magnitude slower than tree based approaches.

From the aspect of coding, ensemble trees can be used as a codebook for efficient encoding [36] or learning better descriptors [37], [38]. Ensemble trees have recently been exploited for direct feature mapping to handle non-linear classification [39], [40]. In this work, we demonstrate the effectiveness of ensemble tree induced features in shape regression.

III. Regressing Local Binary Features

In Equation (1), both the linear regression matrix W^t and the feature mapping function Φ^t are unknown. In our approach, we propose learning them in two consecutive steps. We first learn a local feature mapping function to generate local binary features for each landmark. We concatenate all local features to get Φ^t. Then we learn W^t by linear regression. This learning process is repeated stage-by-stage in a cascaded fashion. Figure 1 shows the overview of our approach.

A. Learning Local Binary Features $\boldsymbol{\Phi^t}$

The feature mapping function is composed of a set of local feature mapping functions i.e., $\Phi^t = [\phi_1^t, \phi_2^t, ..., \phi_L^t]$. We learn each of them independently. The regression target for learning ϕ_l^t is the ground truth shape increment $\Delta\hat{S}^t$:

$$\min_{w^t,\phi_l^t} \sum_{i=1} \|\pi_l \circ \Delta\hat{S}_i^t - w_l^t \phi_l^t \left(I_i, S_i^{t-1}\right)\|_2^2, \tag{2}$$

where $\Delta\hat{S}^t = (\Delta\hat{x}_1^t, \Delta\hat{y}_1^t, ..., \Delta\hat{x}_L^t, \Delta\hat{y}_L^t) \in \mathbb{R}^{2L}$ denotes the coordinates of the offsets for all L facial landmarks. i iterates over all training samples, operator π_l extracts two elements $(2l-1, 2l)$ from the vector $\Delta\hat{S}_i$, and $\pi_l \circ \Delta\hat{S}_i$ is the ground truth 2D-offset of lth landmark in ith training sample.

Algorithm 1 Training of Cascade Face Alignment With Local Binary Features

Input: all training samples (images I_i, ground truth shapes $\hat{S}_i$ and initial shapes S_i^0)
Input: the number of stages T, the number of trees in each stage N, the tree depth D, the number of landmarks L
Output: feature mapping functions Φ^t and global linear regression weights W^t for T stages

1: **for** $t = 1$ to T **do**
2: set for all training samples

$$\Delta\hat{S}_i^t = \hat{S}_i - S_i^{t-1},$$

3: **for** $l = 1$ to L **do**
4: learn a regression random forest of $\lceil N/L \rceil$ trees (local feature mapping function ϕ_l^t) as Section 3.1

$$\phi_l^t = \arg\min_{w^t,\phi_l^t} \sum_{i=1} \|\pi_l \circ \Delta\hat{S}_i^t - w_l^t \phi_l^t \left(I_i, S_i^{t-1}\right)\|_2^2,$$

5: **end for**
6: construct Φ^t by concatenate local feature mapping functions $\{\phi_l^t\}_{l=1}^L$
7: learn the global linear regression as Section 3.2

$$W^t = \arg\min_{W^t} \sum_{i=1}^{N} \|\Delta\hat{S}_i^t - W^t\Phi^t(I_i, S_i^{t-1})\|_2^2 + \lambda||W^t||_2^2,$$

8: update shapes for all training samples

$$S_i^t = S_i^{t-1} + W^t\Phi^t(I_i, S_i^{t-1}),$$

9: **end for**

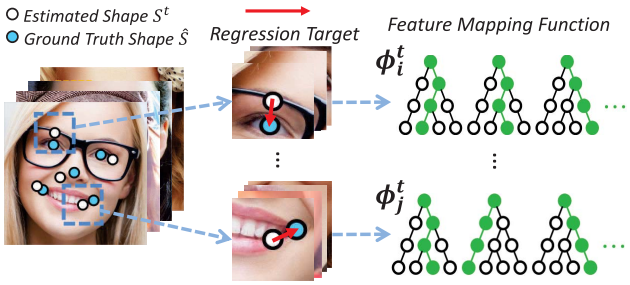


Fig. 2. Learning local feature mapping functions. A random forest is learnt as the feature mapping function for each landmark. For the lth feature mapping function, the local region of the lth landmark is used as input and the corresponding offsets is used as target.

Figure 2 illustrates the process of learning local feature mapping functions on independent landmarks.

We follow the framework proposed by Breiman [41] to train a random forest,[3] using the local region of the lth landmark as the input and the corresponding offsets $\pi_l \circ \Delta\hat{S}_i^t$ as the target. Beginning with the root node, the structure of a tree is constructed by recursively adding binary splitting nodes so that the variance of the regression targets are minimized. The splitting node consists of a pixel difference feature f_θ^l [1], [7]

[3] Although there are advanced methods [1], [42] for training trees, we choose this standard method mainly for two purposes: ensuring fair comparison with other methods; isolating our proposed factors for further analysis.

and a threshold, which are selected from a set of randomly generated candidates.

Given an image I and an estimated shape S, we compute a shape-indexed pixel difference feature f_θ^l in the local region of the lth landmark as follows.

$$f_\theta^l(I, S) = (p_1 - p_2) / (p_1 + p_2), \ \textit{where} \tag{3}$$

$$p_1 = I(\pi_l \circ S + M_S \text{ u}), \tag{4}$$

$$p_2 = I(\pi_l \circ S + M_S \text{ v}) \tag{5}$$

The pixels are refereed to local coordinates $\theta = (\text{u}, \text{v})$ with respect to the lth landmark to ensure shape invariance. Both (u, v) are uniformly sampled from the local region of lth landmarks. To efficiently evaluate the pixels in the original image, the local coordinates are transformed back to the image coordinates by Equations (4,5), where the matrix M_S scales and rotates the local coordinates according to the scale and angle of S relative to a mean shape.

To train each split node, we test 500 randomly sampled features, 10 randomly sampled thresholds per-feature, and pick the feature and correlated threshold that give rise to maximum variance reduction. Testing more features results in only marginal improvement in our experiment. After training, each leaf node stores a 2D offset vector that is the average of all the training samples in the leaf.

We only sample pixel features in a local region around the landmark that is estimated. Using such a local region is critical to our approach. In the training, the optimal region size is estimated in each stage via cross validation. We will discuss more details in Section III-C.

During testing, a sample traverses the trees until it reaches one leaf node for each tree. The output of the random forest is the summation of the outputs stored in these leaf nodes. Supposing the total number of leaf nodes is M, the output can be rewritten as:

$$w_l^t \phi_l^t \left(I_i, S_i^{t-1}\right), \tag{6}$$

where w_l^t is a 2-by-M matrix in which each column is the 2D vector stored in the corresponding leaf node, and ϕ_l^t is a M-dimensional binary vector. For each dimension in ϕ_l^t, its value is 1 if the test sample reaches the corresponding leaf node and 0 otherwise. Therefore, ϕ_l^t is a very sparse binary vector. The number of non-zero elements in ϕ_l^t is the same as the number of trees in the forest, which is much smaller than M. We call such ϕ_l^ts "*local binary features*". Figure 3 illustrates the process of extracting local binary features.

B. Learning Global Linear Regression $\mathbf{W^t}$

After the local random forest learning, we obtain not only the binary features ϕ_l^t, but also the local regression output w_l^t. We *discard* such learned local output w_l^t. Instead, we concatenate the binary features to a global feature mapping function Φ^t and learn a global linear projection W^t by minimizing the following objective function:

$$\min_{W^t} \sum_{i=1}^{N} \| \Delta \hat{S}_i^t - W^t \Phi^t(I_i, S_i^{t-1}) \|_2^2 + \lambda ||W^t||_2^2, \tag{7}$$

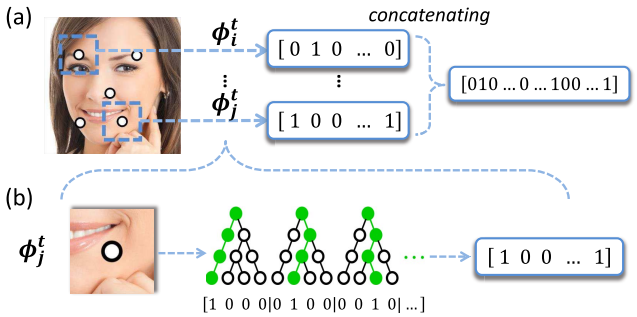


Fig. 3. Local binary features. (a) The local feature mapping function ϕ_l^t encodes the corresponding local region into a binary feature; all local binary features are concatenated to form high-dimensional binary features. (b) We use random forest as the local mapping function. Each extracted binary feature indicates whether the input image contains some local patterns or not.

where the first term is the regression target, the second term is a L2 regularization on W^t, and λ controls the regularization strength. Regularization is necessary because the dimensionality of the features is very high. In our experiment, for 68 landmarks, the dimensionality of Φ^t could be 100K+. Without regularization, we observe substantial overfitting. Because the binary features are highly sparse, we use a dual coordinate descent method [43] to deal with such a large-scale sparse linear system. Since the objective function is quadratic with respect to W^t, we can always reach its global optimum.

We find that such global "relearning" or "transfer learning" significantly improves performance. We believe this is for two reasons. On one hand, the locally learned output by random forest is noisy because the number of training samples in a leaf node may be insufficient. On the other hand, the global regression can effectively enforce a global shape constraint and reduce local errors caused by occlusion and ambiguous local appearance.

C. Locality Principle

As we have described previously, we apply two important regularization methods in feature learning, as guided by a locality principle: 1) we learn a forest for each landmark independently; 2) we only consider the pixel features in the local region of a landmark. In this section, we explain why we made such choices.

Why the local region? We begin with the second choice. Suppose we want to predict the offset Δs of a single landmark and we select features from a local region with radius r. Intuitively, the optimal radius r should depend on the distribution of Δs. If Δs of all training samples are scattered widely, we should use a large r; otherwise we use a small one.

To study the relationship between the distribution of Δs and the optimal radius r, for a landmark we synthesize training and test sample regions whose Δs follow a Gaussian distribution with different standard deviations. For each distribution, we experimentally determine the optimal region radius (in terms of test error) by training regression forests on various radii. We use the same forest parameters (tree depth and number of trees) as in our cascade training. We repeat this experiment

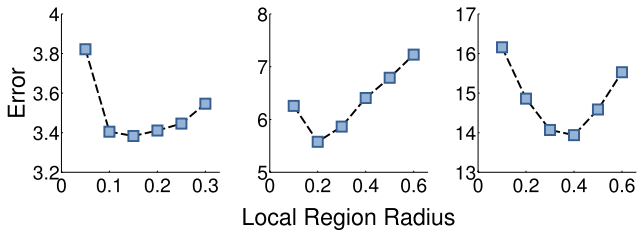


Fig. 4. The horizontal axis stands for the local region radius. The vertical axis stands for the alignment error on a test set. From left to right, standard deviation of the distribution of Δs are 0.05, 0.1, 0.2. Herein both local region radius and alignment error is normalized by the size of face rectangle.

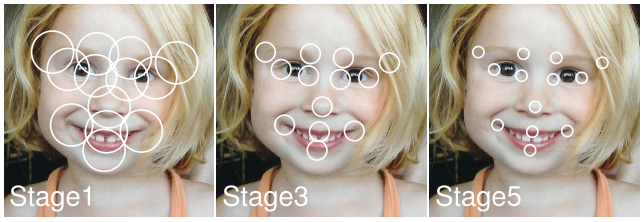


Fig. 5. The best local region sizes at stage 1, 3, and 5.

for all landmarks and take the average of the optimal region radius.

Figure 4 shows the results of three distributions whose std. are 0.05, 0.1, and 0.2 (normalized distance by face rectangle size). The optimal radiuses are 0.12, 0.21 and 0.39. The results indicate that the optimal region radius is almost linearly to the standard deviation of Δs. Therefore, we can conclude that, given limited computation budget (the number of features tested in training forests), it is more effective to only consider candidate features in a local region instead of the global face image.

In our cascade training, at each stage, we search for the best region radius (from 10 discrete values) by cross-validation on a hold-out validation set. Figure 5 shows the best region radiuses found at stage 1, 3, and 5. As expected, the radius gradually shrinks from early stage to later stage, because the variation of regressed face shapes decreases during the cascade.

Why a single landmark regression? It may appear that independent regression of each landmark is sub-optimal. For example, we could probably miss a good feature that can be shared by multiple landmarks. However, we argue that local regression has a few advantages over global learning such as in [1].

First, the feature pool in local learning is less noisy. There may be more useful features in global learning. But the "signal-to-noise ratio" in global learning could be lower, which will make feature selection more difficult.

Second, using local learning does not mean that we do local prediction. In our approach, the linear regression in the second step exploits all learned local features to make a global prediction. Because the local learning of landmarks is independent, the resulting features are by nature more diverse and complementary to each other. Such features are more appropriate for global learning in the second step.

Last, the local learning is adaptive in different stages. In the early stage, the local region size is relatively large and a local region actually covers multiple landmarks. The features learned from one landmark can indeed help its neighboring landmarks. In the late stage, the region size is small and local regression fine-tunes each landmark. Local learning is actually more appropriate in the late stage.

Note that we do not claim that global learning is inferior to our local learning by nature. We believe that local learning delivers better performance mainly for practical reasons. Given limited training capability (the amount of training data, affordable training time, available computing resources, and power of learning algorithm), the local approach can better resist noisy features in the global feature pool, which is extremely large and may cause over fitting. We hope our empirical findings in this work can encourage more similar investigations in the future.

IV. Implementation Details

Our algorithm has two working scenarios. One is single image alignment, which is initialized with the output rectangles of a face detector. Another is face alignment/tracking in video sequences, which is initialized by the alignment output shape of the previous frame. In these two scenarios, all components of our method are generic except two of them which are specifically designed. One is the initialization shape in testing and another is the corresponding data augmentation method in training. We will discuss these two parts and some important implementation details in this section.

A. Initial Shape

In our method, the transformed mean shape is used as the initial shape. In single image alignment, the output face rectangle of a face detector is used to estimate the transformation. The quality of the detector greatly affects the alignment performance in this scenario. We will study this issue in Section VI. In face tracking, the transformation is estimated with the aligned shape from the previous frame. Compared with per frame initialization with a face detector, initialization with the previous frame makes face tracking more stable due to the more accurate shift, scale, and rotation information inherited from the pervious frame.

For video tracking, directly using the predicted shape from the pervious frame seems to be straightforward. But in experiments, we found that initialization with transformed mean shape has better performance. The possible reason is that our tracking model is trained with static images but not labeled video sequences. Estimating the distribution of face motion from static images is a hard problem and may cause large errors.

B. Data Augmentation

Our training data contains three parts: images, ground truth shapes and initial shapes. We augment our training data with multiple initializations for one image. Data augmentation serves as an effective method for improving the generation of training. In our implementation, we use an augmentation of 50 times or more for training the linear projection and randomly select 10% of the augmented training data for

Algorithm 2 Data Augmentation

Input: ground truth shapes[4] $\hat{S}_i$, detected face rectangles B_i
Input: standard rectangle B
Input: data augmentation times K
Output: initial shapes S_{ik}^0

1: **for** all training samples **do**
2: get rectangle geometric transformation T_{1i}

$$T_{1i} = \arg\min_T \|T \cdot B_i - B\|_2^2,$$

3: get normalized ground truth shape $\widetilde{S}_i$

$$\widetilde{S}_i = T_{1i} \cdot \hat{S}_i,$$

4: **end for**
5: get mean shape $\bar{S}$ with all normalized ground truth shapes

$$\bar{S} = \frac{1}{N}\sum_{i=1}^{N} \widetilde{S}_i,$$

6: **for** all training samples **do**
7: get shape transformation T_{2i}

$$T_{2i} = \arg\min_T \|T \cdot \widetilde{S}_i - \bar{S}\|_2^2,$$

8: **end for**
9: **for** k = 1 to K **do**
10: random shuffle $\{T_{2i}\}$ to $\{T_{2i'}\}$
11: **for** all training samples **do**
12: calculate initial shapes

$$S_{ik}^0 = T_{1i}^{-1} \cdot T_{2i'}^{-1} \cdot \bar{S},$$

13: **end for**
14: **end for**

training random forests of each landmark to speed up the training phase.

In order to augment the data, we first model the relative position distribution between the face rectangle and the ground truth landmarks, and then sample from the distribution to position the mean shape around the ground truth shape. Algorithm 2 shows the data augmentation algorithm for single image alignment. For face tracking, the only difference is that the transformation T_{1i} in Step 2 is estimated by aligning the ground truth shape to the mean shape.

Similarity (4 degrees of freedom) and affine (6 degrees of freedom) are both feasible candidates for the geometric transformation in data augmentation. But for face tracking, since shear transformation is rare, we recommend anisotropic scaling similarity transform (5 degrees of freedom) which makes alignment more robust in our experiments.

C. Normalizing Regression Target

Although it is straight forward to use the delta shape $\hat{S} - S^{t-1}$ as the regression target, we suggest using the normalized delta shape as the regression target. The reason is that it will effectively reduce variations due to scale and rotation, resulting in easier regression task and better performance. In practice, we first apply the inverse geometric transform of S^{t-1} to normalize $\hat{S}$ and S^{t-1}, and then use the difference between the normalized shapes as the regression target.

In testing, instead of directly adding the predicted shape increment to S^{t-1}, we first scale and rotate (without translating) the predicted shape increment according to the geometric transform of S^{t-1}, and then add the transformed shape increment to S^{t-1}.

D. Learning the Linear Projection

The dimensionality of the binary feature can be very high (100K+) and the number of training data after augmentation can also be very large (50K+). Practical issues may arise in solving such a large linear system. As our binary features are highly sparse, we use a dual coordinate descent method [43] to deal with such large-scale sparse data. To avoid overfitting, an additional L2 penalty term is added to the original objective function, Equation 7 to regularize the linear projection.

E. Efficient Linear Projection

Direct linear projection on the binary features via generic matrix multiplication is inefficient. The linear projection can be dramatically sped up by taking the advantages of sparse and binary properties of our features. In practice, we compute the linear projection by directly adding the columns of W^t which correspond to non-zero dimensions of the binary feature. This avoids redundant computation on zeros and the expensive multiplication operations, leading to extremely fast testing speed.

F. Parameter Settings

Our approach has a few free parameters: the number of stages T, the total number of trees in each stage[5] N, and the tree depth D. To test different speed-accuracy trade-offs, we use two sets of settings: 1) more accurate: $T = 5$, $N = 1200$, $D = 7$; and 2) faster: ($T = 5, N = 300, D = 5$). We call the two versions *LBF* (local binary features) and *LBF fast*. Besides these two proposed settings, in Section V-C, we will explore the parameters in details to validate the parameter selection.

G. Evaluation Metric

Following the standard [1], [31], we use the inter-pupil distance normalized landmark error. For datasets (e.g. *300-W*) which have no pupil landmark labeled, we use average position of the landmarks around the eye instead. For each dataset we report the error averaged over all landmarks and images. Note that the error is represented as a percentage of the pupil-distance, and we drop the notation % in the reported results for clarity.

[4] All shapes in this algorithm are in homogeneous coordinate system for concise description of the geometric transform.

[5] We fix the total number of trees so few trees will be used for each landmark if there are more landmarks.

V. Experiments of Alignment

Datasets: There are quite a few datasets for face alignment. We use three more recent and challenging ones. They present different variations in face shape, appearance, and number of landmarks.

LFPW (29 landmarks) [31] is collected from the web. As some URLs are no longer valid, we only use 717 of the 1,100 images for training and 249 of the 300 images for testing. Although each image is labeled with 35 landmarks, we use 29 of the 35 landmarks in our experiments, following previous work [1].

Helen (194 landmarks) [44] contains 2,300 high resolution web images. We follow the same setting in [44]: 2000 images for training and 330 images for testing. The high resolution is beneficial for high accuracy alignment, but the large number of landmarks is challenging in terms of computation.

300-W (68 landmarks) is short for 300 Faces in-the-Wild [20]. It is created from existing datasets, including LFPW [31], AFW [18], Helen [44], XM2VTS [45], and a new dataset called IBUG. It is created as a challenge and only provides training data. We split their training data into two parts for our own training and testing. Our training set consists of AFW, the training sets of LFPW, and the training sets of Helen, with 3148 images in total. Our testing set consists of IBUG, the testing sets of LFPW, and the testing sets of Helen, with 689 images in total. We do not use images from XM2VTS as it is taken under a controlled environment and is too simple. We should point out that the IBUG subset is extremely challenging as its images have large variations in face poses, expressions and illuminations.

Our method needs face rectangles to initialize alignment. For 300-W, we use the official provided rectangles. For LFPW and Helen which have no official rectangles, we use a multi-pose V.J. detector. We will discuss more about the detector and the method way we generate face rectangles in Section VI.

In the following section, we first compare our approach against state-of-the-art methods, then validate the proposed approach via comparison with certain baseline methods.

A. Comparison With State-of-the-Art Methods

During our training, we use similar data augmentation as in [1] to enlarge the training data and improve generalization ability. Note that during testing we only use the mean shape as the initialization. We do not use multiple initializations and median based refinement as in [1].

Explicit shape regression (ESR) [1] and the supervised descent method (SDM) [6] are two of the best methods for face alignment. We implement these two methods and our implementation achieves comparable accuracy to that which was reported by the original authors. For comparison with other methods, we use the original results in the literature. Table I and Table II report the errors and speeds (frames per second or FPS) of all compared methods on three datasets. Note that we also divide the testing set of 300-W into two subsets: the common subset consists of the testing sets of Helen and LFPW, and the challenging IBUG subset. We report all results on the two subsets as well.

TABLE I

Error and Runtime (in FPS) on LFPW, Helen and 300-W Datasets, Respectively. The Errors of ESR and SDM Are From Our Implementation. Note That ESR and SDM Have Reported Error on LFPW in the Original Papers. Their Accuracy Is Similar as Ours (3.43 and 3.47, Respectively)

LFPW (29 landmarks)			Helen (194 landmarks)		
Method	Error	FPS	Method	Error	FPS
[31]	3.99	≈ 1	STASM[46]	11.1	-
ESR[1]	3.47	220	CompASM[44]	9.10	-
RCPR[7]	3.50	-	ESR[1]	5.70	70
SDM[6]	3.49	160	RCPR[7]	6.50	-
EGM[32]	3.98	< 1	SDM[6]	5.85	21
LBF	**3.35**	460	**LBF**	**5.41**	200
LBF fast	**3.35**	**4200**	**LBF fast**	5.80	**1500**

TABLE II

Error and Runtime (in FPS) on 300-W Datasets. The Errors of ESR and SDM Are From Our Implementation. * Indicates That This Method Uses Extra Training Data Besides 300-W

300-W (68 landmarks)[6]				
Method	Fullset	Common Subset	Challenging Subset	FPS
RCPR[7]	8.35	6.18	17.26	80
CFAN[11]	7.69	5.50	16.78	44
ESR[1]	7.58	5.28	17.00	120
SDM[6]	7.52	5.60	15.40	70
ERT[9]	6.40	-	-	1000
CFSS[47]	5.76	4.73	9.98	25
TCDCN[12]*	**5.54**	4.80	**8.60**	56
LBF	6.32	4.95	11.98	320
LBF fast	7.37	5.38	15.50	**3100**
LBF + box refine	5.85	**4.63**	10.83	210

1) Comparison of Accuracy: Overall, the regression-based approaches are significantly better than ASM-based methods. From the aspect of feature, learnt pixel based methods [1], [7], [9] (including our method) and SIFT based methods [6], [47] achieve very good performance on the task of face alignment. CNN based methods [11], [12] have shown great potential and achieved even better performance on the task of face alignment.

Compared with former best methods, ESR and SDM, our method achieves significant error reduction with respect to ESR and SDM of 30% and 22%, respectively, on the challenging IBUG subset. We believe this is due to the good generalization ability of our method. ERT [9] is a concurrency work with our conference version. It is based on boosting trees, similar to ESR. Smart usage of shrinkage and exponential prior in feature selection brings large improvement to ERT compared with ESR. The improvement of ERT is orthogonal to the improvement of our approach, opening opportunity for better methods.

It is interesting to note that our method can achieve better accuracy than SDM which is based on SIFT. This could be for

[6]To keep consistent with LFPW and Helen, here we still use landmark error normalized by the inter-pupil distance, but not normalized by the Euclidean distance between the outer corners of the eyes (which is recommended in 300-W website).

several reasons. First, although our pixel features are simpler than SIFT, they are specifically learnt for the task while SIFT is manually designed and general. We believe our new learning scheme (local learning + global refinement) is the key to achieving a high level of accuracy using such simple pixel features. Secondly, our local binary feature benefits from the higher dimension compared with SIFT. Thirdly, we follow the same settings in the original SDM paper. Multi-scale SIFT and denser extraction may bring further accuracy improvement for SDM. Overall, our focus is the new learning scheme and its state-of-the-art performance (especially high speed).

Recently, deep learning based methods [11], [12] bring considerable progress on face alignment. Besides the powerful CNN feature, many of these methods estimate the face shape firstly instead of directly applying cascade shape regression, to reduce the negative impact from un-stable face detector. In comparison with these methods, we also develop a similar component to find a better initialization. Instead of estimating the face shape directly, we use local sliding window based alignment method to find a better face localization for initialization. Specially, we use a face detector (JCDA [19]) based on the proposed alignment method to scan a small area around the original face rectangle to find a better face rectangle. We denote this version as 'LBF + box refine'. Good initialization greatly reduces the performance gap between our method and deep learning based alignment, considering that our approach is much faster. We will discuss more about the effect of good initialization in Sec.VI

2) Comparison of Speed: Our approach, ESR, and SDM are all implemented in C++ and tested on a single core i7-2600 CPU. The speed of other methods is quoted from the original papers. While ESR and SDM are already the fastest face alignment methods in the literature, our method has an even larger advantage in terms of speed. Our fast version is dozens of times faster and achieves thousands of FPS for a large number of landmarks. The high speed comes from sparse binary features. As each testing sample has only a small number of non-zero entries in its high dimensional features, the shape update is performed only a few times with an efficient look up table and vector addition, instead of matrix multiplication in the global linear regression. The surprisingly high performance makes our approach especially attractive for applications with limited computational power. For example, our method runs in about 300 FPS on a mobile. This opens up new opportunities for online face applications on mobile phone.

B. Validation of Proposed Approach

We verify the effectiveness of the two key components of our approach, *local learning* and *binary features*, by comparing them with baseline methods that only differ in those aspects but remain exactly the same in all others. We use the 300-W dataset and LBF settings.

1) Local Learning vs. Global Learning: In the baseline method, the difference is that, during the learning of local binary features, the pixels are indexed over the global shape, in the same way as [1], instead of only in a local region around the local landmark as in the proposed approach. Regression is performed on the entire shape instead of only the local landmark. All other parameters are the same to ensure the same training effort. We call this baseline *global learning*. Figure 6 shows that the proposed *local learning* is significantly better (25% error reduction) and verifies that it is capable of finding much better features.

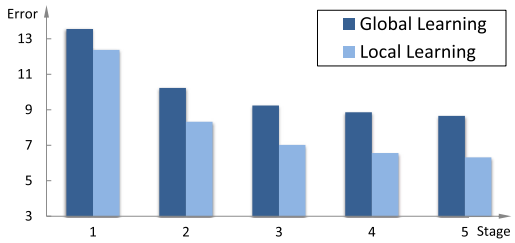


Fig. 6. Comparison between local learning and global learning.

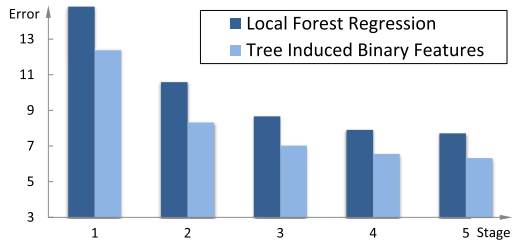


Fig. 7. Comparison between tree induced binary features and local forest regression.

2) Tree Induced Binary Features vs. Local Forest Regression: In the baseline method, we do not use the locally learned high dimensional binary features for global regression. Instead, we directly use the local random forest's regression output (a 2D offset vector) of each landmark as features to learn a global regression in the same way. Note that the learning process of the local trees is also exactly the same. Figure 7 shows that high dimensional binary features clearly outperform the simple raw output from local regression as features, because the former faithfully retains the full information of local learning.

C. Parameter Settings

There are three important free parameters of our approach: the number of stages T, the total number of trees in each stage N, and the tree depth D. To deeply understand our method and explore the relationship between performances, the computational cost and the storage cost, we vary the parameters and report the performance in Figures 8 and 9. In these experiments, we use the 300-W dataset.

The computational complexity of our method in the testing phase is $O(T * N * D + T * N * L)$. The first part originates from the prediction of local binary features and the linear projection contributes to the second part. L (the number of landmarks) is fixed when a trained alignment model is given. We note that the first part dominates the computational cost in our implementation with $D >= 3$. So the computational complexity is approximated to $O(T * N * D)$. The storage

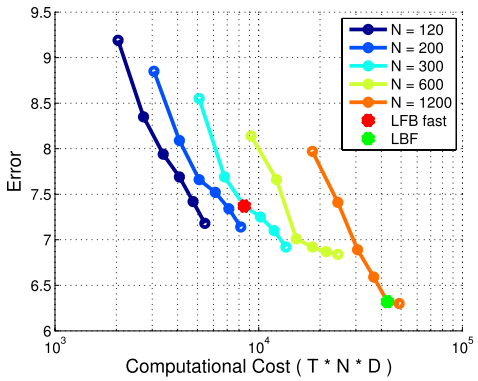


Fig. 8. Relationship between alignment error and computational cost varying parameter settings. In legends, N indicates the total number of trees in each stage. For each curve (each N), the tree depth varies from 3 to 8. The number of stages is fixed to 5.

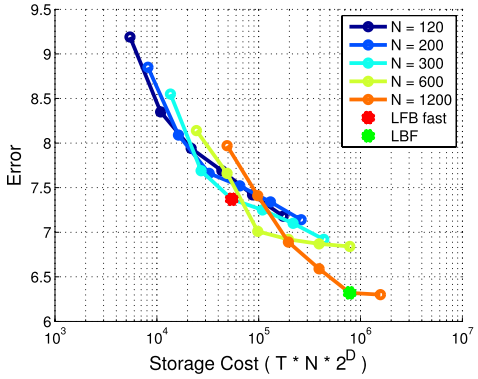


Fig. 9. Relationship between alignment error and storage cost varying parameter settings. In legends, N indicates the total number of trees in each stage. For each curve (each N), the tree depth varies from 3 to 8. The number of stages is fixed to 5.

cost of our method is $O(T * N * 2^D * L)$, which is associated with the delta shape stored in each tree leaf node.

In Figures 8 and 9, we vary N and D (fixing $T = 5$) and show the performance. These two plots indicate that our LBF fast model is the most compact model at a given performance level, through its performance is not the best when the constant computational cost is given. We hope such performance analyses could be helpful in the performance and computational/storage cost tradeoff on our method.

VI. Benefit From Good Initialization

There are two sources of challenges for face alignment. The first is the large variance in the facial appearance, originating from partial occlusions, lightings and noises. The second is the large variance in initial face shapes, derived from face detection results. Previous learning based alignment methods are mostly focusing on the first challenge but pay little attention to the second one.

In this work, we perform the first quantitative study on how face detector quality can affect alignment accuracy and discuss how to obtain better initialization. In Section 6.1, we first show that synthesized "good initialization" can produce significant improvement in alignment accuracy, indicating that choosing a good detector is of great importance for alignment. In Section 6.2, we compare a few off-the-shelf face detectors under two scenarios. We find that an "alignment friendly" detector significantly outperforms others and further improves our state-of-the-art results. In Section 6.3, we propose a simple yet effect metric for such "alignment friendliness" of a detector, which facilitates practical applications. We hope our new experiment and findings can motivate future research on joint alignment and detection.

TABLE III

Alignment Performance Initialized With 300-W Official Detector and Pseudo Ground Truth Rectangles. The Pseudo Ground Truth Rectangles Are Regressed From Labeled Face Landmarks

Method	Official Detector	Pseudo ground truth Rectangles	Relative Improvement
LBF	6.32	5.72	9.5%
LBF fast	7.37	6.04	18.0%

A. Potential Improvement From Good Initialization

To explore potential gain from reducing the variance in initialization, we come up with a constructor of face rectangles considering the labeled landmarks. Given 5 face landmarks (left/right eyes center, left/right mouth center, and tip of nose), a linear regression function is learnt to predict the center and scale of the rectangle. We note that the learning target is the detected face rectangles from 300-W official detector, thus the regression is not optimal and just functions as rectification and smoothness. The predicted rectangles are nominated as "pseudo ground truth face rectangles".

The pseudo ground truth rectangles are significantly better than normal face detectors. On the 300-W dataset, compared with the official "in-house detector",[7] the pseudo ground truth rectangles can bring up to 18% relative error reduction, which is shown in Table III. The improvement from the pseudo ground truth rectangles is notable and essential. It can reduce the difficulty of the learning problem with alignment. For our method, with these rectangles, LBF fast model surpasses LBF model with 300-W official rectangles, and our LBF model can be much stronger.

The key information of the pseudo ground truth rectangles is the location of face components. Although the pseudo ground truth rectangles based on labeled landmarks is not practical in reality, it shows us the potential improvement. The potential improvement can be obtained by a detector which also considers the location of the face components in practice.

B. Practical Improvement From Alignment-Friendly Detector

To obtain the potential improvement in practice, we will consider a detector which is friendly to alignment, compare it with several detectors and show the improvement.

Joint cascade detection and alignment detector (JCDA) [19] is an efficient face detector inspired by the Viola-Jones style

[7] rectangles marked as "bb_detector" in http://ibug.doc.ic.ac.uk/media/uploads/competitions/bounding_boxes.zip

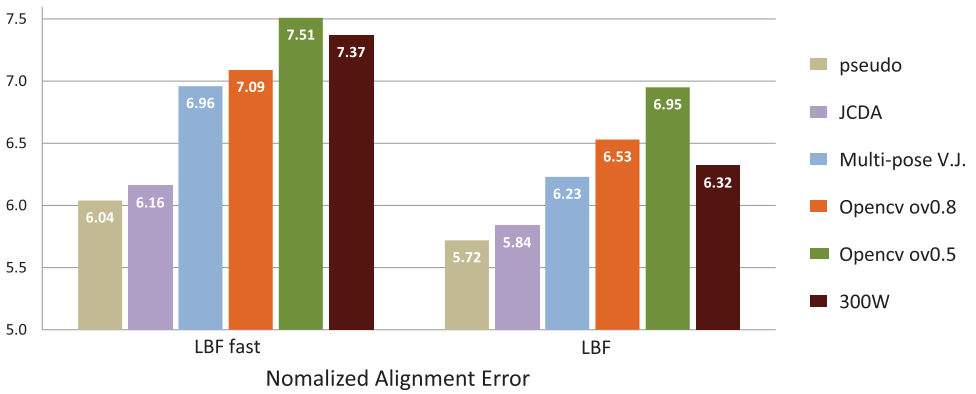


Fig. 10. Alignment performance based on different detectors under alignment research scenario. For each detector, alignment performance of LBF fast model(left) and LBF model(right) are shown. 300-W Fullset of test is used for test. An alignment-friendly detector can greatly improve the alignment performance, and the improvement by better detector can be even larger than changing LBF fast model to LBF model (the latter one has 16 times more compute cost). (Best viewed in color).

detectors [48] and the proposed alignment method. It uses alignment information to facilitate detection, so its output face rectangles are "aligned" with the face landmarks. This "aligned" property similar to the pseudo ground truth rectangles is of great benefit to the alignment method. Although this detector can achieve landmarks directly, to pursue better performance and to compare with other detectors, we only use its output face rectangles, followed by an extra alignment phase in our experiments.

Besides the JCDA detector which is friendly to alignment, two widely used detectors are also used for comparison. *OpenCV detector* is a cascade face detector based on the Viola-Jones style detector of [48] and [49]. This detector is trained with small scale images and Haar-like features. It has high recall but tends to generate false alarms and low quality face rectangles.[8] *Multi-pose V.J. detector* is a well trained Viola-Jones style commercial detector [48] with a large amount of training images and contains a well trained post-filter. Multiple classifiers are used for different poses. Overall, this detector has less false alarms and less poor aligned rectangles compared with OpenCV's detector.

To validate the gain from alignment-friendly detector, we make evaluations under two scenarios.

1) Alignment Evaluation Scenario: This scenario evaluates the alignment algorithm without considering the poor quality output of the face detector, which is adopted by most alignment papers. In this scenario, the effect of false alarms, missed detections and poorly aligned rectangles of face detectors are eliminated. Specially, we filter the output rectangles of the detector by selecting the ones which have the largest overlap with labeled faces and by dropping redundant ones. Moreover, for OpenCV's detector, which has many poorly aligned rectangles, we drop rectangles which have smaller overlap than a particular threshold. The calculated overlap is normalized by the ground truth bounding box. We adopt two thresholds for the dropping – 0.5 and 0.8, which are marked as OpenCV ov0.5 and OpenCV ov0.8 in Figure 10.

[8]We use model "haarcascade frontalface alt" provided by OpenCV, for it has less false alarms compared with the default model.

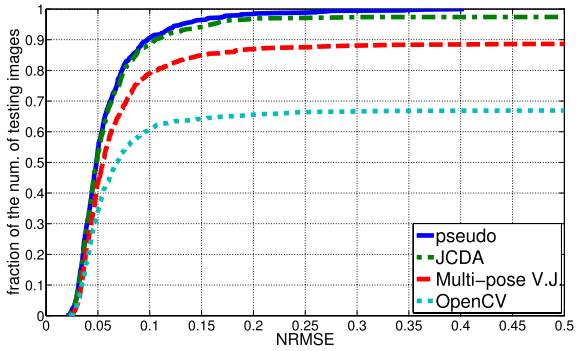


Fig. 11. Alignment performance based on different detectors under a practical application scenario. LBF fast setting is used as the alignment for all detectors and 300-W Fullset of test is used for the test. (Best viewed in color).

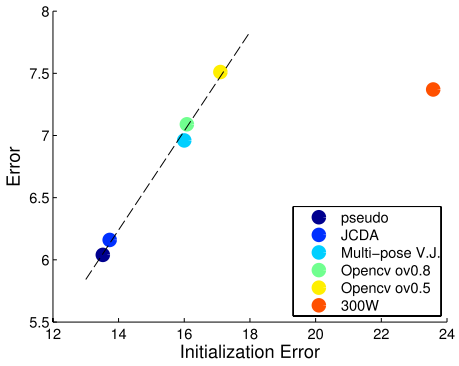


Fig. 12. Relationship between the initialization error after data augmentation (Algorithm 2) and the final alignment error. All detectors output square face rectangles except 300W official detector notated as "300W".

Figure 10 shows the alignment performance on the 300-W dataset with several detectors. Compared with the performance of the 300-W official detector (notated as 300W in Figure 10 and reported in Section V-A), the JCDA detector further improves our LBF fast model by 16% relatively, which is even larger than the promotion from LBF fast model to LBF model. The improvement of the JCDA detector indicates that the potential gain with pseudo ground truth rectangles can

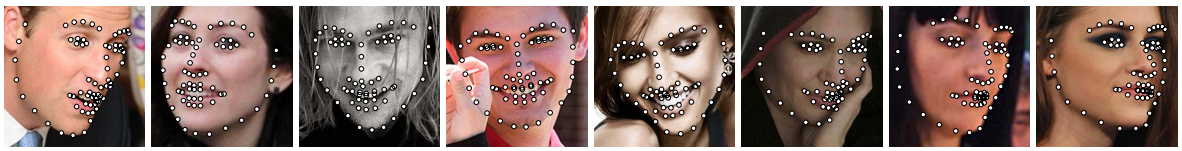


Fig. 13. Example results from the Challenging Subset of the 300-W dataset.

Fig. 14. Example results from the Helen dataset.

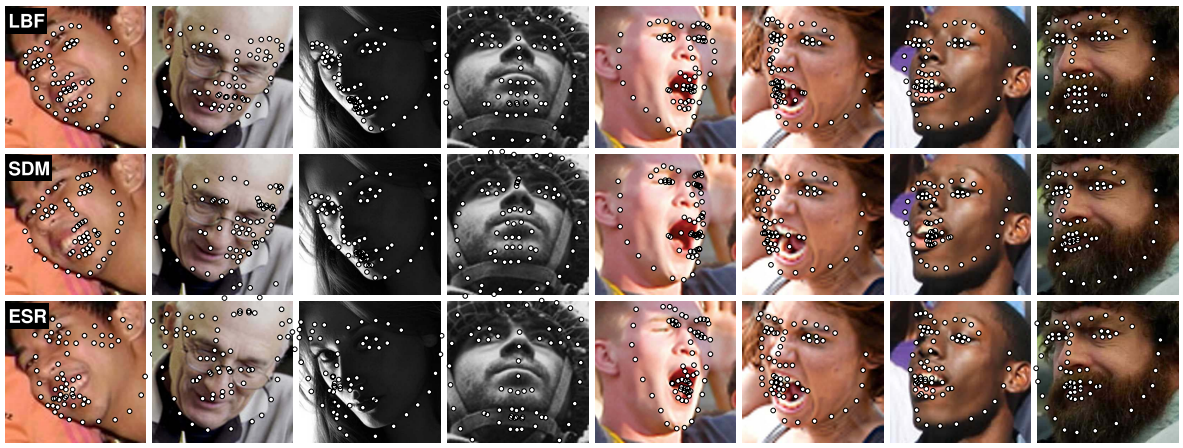


Fig. 15. Some failure cases from the Challenging Subset of 300-W dataset.

be achieved with an alignment-friendly detection algorithm in practical application.

2) Practical Application Scenario: In this scenario, to estimate the whole system performance of detection and alignment, the cases dropped by the alignment evaluation scenario are added back in for consideration. All detected faces are aligned and scored. To punish detectors with low recall, a maximum alignment error is added to total error for each missed face. On the 300-W testset, since not all faces are labeled, we manually mark all un-labeled faces to construct a black list. Once a detected face has more than 0.2 overlap with any face in the black list, it is ignored.

Figure 11 shows the alignment error curve with several detectors under the practical application scenario. Specially, none of the overlap threshold filters are used, so there is only one curve for OpenCV detection. The dataset is selected based on 300-W official detector, thus this detector could not be evaluated under this scenario. We can notice that in Figure 11, the difference between detectors is larger than the one shown in Figure 10. We think Figure 11 is a more realistic reflection of the system performance improvement. In [19], it is shown that alignment can help detection for a better performance. Here we show a detector, which considers alignment and uses alignment information, can in turn improve alignment. A joint optimization of detection and alignment may bring further improvement in this field.

C. A Metric of "Alignment Friendliness" for Detectors

As shown, an alignment-friendly detector is of great importance for the accuracy of alignment. Nevertheless, it is still unclear how to simply measure such "friendliness" of an off-the-shelf detector. In Section 6.2, comprehensive experiments have been done to compare a few detectors. This is costly (training and testing a lot of models) and inconvenient for practical scenarios, e.g., how to choose a detector from a few candidates, or how to tune the detector's parameters, such that the detector works best for subsequent alignment.

We propose a simple metric of "alignment friendliness". It is derived based on the intuition that alignment accuracy is correlated to how well the initial face rectangle is aligned to the true face shape, that is, the relative position distribution between the initial face detection rectangle and the ground truth landmarks is highly correlated to the final alignment accuracy. We note that such distribution has already been modeled in our data augmentation part (Algorithm 2). Thus, we can use the initialization error (discrepancy between initial face shapes and ground truth) after data augmentation as the metric. As a validation, in Figure 12 we observe that the final alignment error is highly correlated with the initialization error

on most detectors. We note that the 300W official detector is an outlier because it outputs non-square face rectangles while all others output square face rectangles.

The proposed metric is simple to compute and highly indicative of a detector's quality for alignment. We hope it would be useful in selecting and tuning detection and alignment methods.

VII. Limitations

LBF is a very efficient method for face alignment, thanks to the rapid pixel based feature. However, also limited by its pixel based feature, it is more sensitive to image noise compared with SIFT and CNN based methods. Using harr-like pixel based feature to enhance its robustness is one of our future works.

Modern face alignment methods work well on most cases, but for extremely difficult samples (Figure 15) which mix large head poses, extreme lighting, and partial occlusions, there are still many failure cases. This is partially because of lacking of training data. A large training set as difficult as IBUG should be collected. On the other hand, alignment algorithm could be improved to handle these hard cases.

VIII. Conclusion

In this work, we have presented a novel approach to learning local binary features for highly accurate and extremely fast face alignment. The shape regression framework regularized by locality principle is also promising for use in other relevant areas such as anatomical structure segmentation and human pose estimation. Furthermore, it is worth exploring the refitting strategy in other scenarios where regression trees are applied.

Through experiments, we find an alignment-friendly detector based on our method can further improve the proposed approach significantly. This suggests that joint consideration of the detector and alignment is worthy of further study.

References

[1] X. Cao, Y. Wei, F. Wen, and J. Sun, "Face alignment by explicit shape regression," *Int. J. Comput. Vis.*, vol. 107, no. 2, pp. 177–190, 2014.

[2] M. Dantone, J. Gall, G. Fanelli, and L. Van Gool, "Real-time facial feature detection using conditional regression forests," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2012, pp. 2578–2585.

[3] P. Dollar, P. Welinder, and P. Perona, "Cascaded pose regression," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2010, pp. 1078–1085.

[4] M. Valstar, B. Martinez, X. Binefa, and M. Pantic, "Facial point detection using boosted regression and graph models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2010, pp. 2729–2736.

[5] C. Cao, Y. Weng, S. Lin, and K. Zhou, "3D shape regression for real-time facial animation," *ACM Trans. Graph.*, vol. 32, no. 4, 2013, Art. ID 41.

[6] X. Xiong and F. De la Torre, "Supervised descent method and its applications to face alignment," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2013, pp. 532–539.

[7] X. P. Burgos-Artizzu, P. Perona, and P. Dollár, "Robust face landmark estimation under occlusion," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2013, pp. 1513–1520.

[8] Y. Sun, X. Wang, and X. Tang, "Deep convolutional network cascade for facial point detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2013, pp. 3476–3483.

[9] V. Kazemi and J. Sullivan, "One millisecond face alignment with an ensemble of regression trees," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2014, pp. 1867–1874.

[10] J. Xing, Z. Niu, J. Huang, W. Hu, and S. Yan, "Towards multi-view and partially-occluded face alignment," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2014, pp. 1829–1836.

[11] J. Zhang, S. Shan, M. Kan, and X. Chen, "Coarse-to-fine auto-encoder networks (CFAN) for real-time face alignment," in *Computer Vision*. Springer, 2014, pp. 1–16.

[12] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, "Facial landmark detection by deep multi-task learning," in *Computer Vision*. Springer, 2014, pp. 94–108.

[13] T. Sheerman-Chase, E.-J. Ong, and R. Bowden, "Non-linear predictors for facial feature tracking across pose and expression," in *Proc. 10th IEEE Int. Conf. Workshops Autom. Face Gesture Recognit. (FG)*, Apr. 2013, pp. 1–8.

[14] E.-J. Ong, Y. Lan, B. Theobald, R. Harvey, and R. Bowden, "Robust facial feature tracking using selected multi-resolution linear predictors," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Sep. 2009, pp. 1483–1490.

[15] T. F. Cootes, M. C. Ionita, C. Lindner, and P. Sauer, "Robust and accurate shape model fitting using random forest regression voting," in *Computer Vision*. Springer, 2012, pp. 278–291.

[16] J. Yan, Z. Lei, D. Yi, and S. Z. Li, "Learn to combine multiple hypotheses for accurate face alignment," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops (ICCVW)*, Dec. 2013, pp. 392–396.

[17] H. Li, G. Hua, Z. Lin, J. Brandt, and J. Yang, "Probabilistic elastic part model for unsupervised face detector adaptation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2013, pp. 793–800.

[18] X. Zhu and D. Ramanan, "Face detection, pose estimation, and landmark localization in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2012, pp. 2879–2886.

[19] D. Chen, S. Ren, Y. Wei, X. Cao, and J. Sun, "Joint cascade face detection and alignment," in *Computer Vision*. Springer, 2014, pp. 109–122.

[20] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic, "A semi-automatic methodology for facial landmark annotation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2013, pp. 896–903.

[21] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearance models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 6, pp. 681–685, Jun. 2001.

[22] I. Matthews and S. Baker, "Active appearance models revisited," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 135–164, 2004.

[23] L. Liang, R. Xiao, F. Wen, and J. Sun, "Face alignment via component-based discriminative search," in *Computer Vision*. Springer, 2008, pp. 72–85.

[24] R. Gross, I. Matthews, and S. Baker, "Generic vs. person specific active appearance models," *Image Vis. Comput.*, vol. 23, no. 12, pp. 1080–1093, 2005.

[25] R. Gross, I. Matthews, and S. Baker, "Active appearance models with occlusion," *Image Vis. Comput.*, vol. 24, no. 6, pp. 593–604, 2006.

[26] P. Sauer, T. F. Cootes, and C. J. Taylor, "Accurate regression procedures for active appearance models," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, 2011, pp. 1–11.

[27] P. A. Tresadern, P. Sauer, and T. F. Cootes, "Additive update predictors in active appearance models," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, 2010, pp. 91.1–91.12.

[28] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham, "Active shape models—Their training and application," *Comput. Vis. Image Understand.*, vol. 61, no. 1, pp. 38–59, 1995.

[29] D. Cristinacce and T. F. Cootes, "Feature detection and tracking with constrained local models," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, 2006, pp. 95.1–95.10.

[30] D. Cristinacce and T. F. Cootes, "Boosted regression active shape models," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, 2007, pp. 1–10.

[31] P. N. Belhumeur, D. W. Jacobs, D. J. Kriegman, and N. Kumar, "Localizing parts of faces using a consensus of exemplars," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2011, pp. 545–552.

[32] F. Zhou, J. Brandt, and Z. Lin, "Exemplar-based graph matching for robust facial landmark localization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2013, pp. 1025–1032.

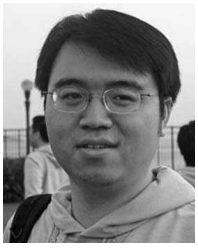
[33] B. M. Smith and L. Zhang, "Joint face alignment with non-parametric shape models," in *Computer Vision*. Springer, 2012, pp. 43–56.

[34] Y. Wang, S. Lucey, and J. F. Cohn, "Enforcing convexity for improved alignment with constrained local models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2008, pp. 1–8.
[35] J. M. Saragih, S. Lucey, and J. F. Cohn, "Deformable model fitting by regularized landmark mean-shift," *Int. J. Comput. Vis.*, vol. 91, no. 2, pp. 200–215, Jan. 2011.
[36] F. Moosmann, B. Triggs, and F. Jurie, "Fast discriminative visual codebooks using randomized clustering forests," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 1–7.
[37] Z. Cao, Q. Yin, X. Tang, and J. Sun, "Face recognition with learning-based descriptor," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2010, pp. 2707–2714.
[38] W. Zhang, X. Wang, and X. Tang, "Coupled information-theoretic encoding for face photo-sketch recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2011, pp. 513–520.
[39] C. Vens and F. Costa, "Random forest based feature induction," in *Proc. 11th IEEE Int. Conf. Data Mining (ICDM)*, Dec. 2011, pp. 744–753.
[40] M. Kobetski and J. Sullivan, "Discriminative tree-based feature mapping," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, 2013, pp. 71.1–71.11.
[41] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
[42] M.-T. Pham and T.-J. Cham, "Fast training and selection of haar features using statistics in boosting-based face detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2007, pp. 1–7.
[43] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *J. Mach. Learn. Res.*, vol. 9, pp. 1871–1874, Jun. 2008.
[44] V. Le, J. Brandt, Z. Lin, L. Bourdev, and T. S. Huang, "Interactive facial feature localization," in *Proc. 12th Eur. Conf. Comput. Vis. (ECCV)*, 2012, pp. 679–692.
[45] K. Messer, J. Matas, J. Kittler, J. Luettin, and G. Maitre, "XM2VTSDB: The extended M2VTS database," in *Proc. 2nd Int. Conf. Audio Video-Based Biometric Person Authentication*, 1999, pp. 1–6.
[46] S. Milborrow and F. Nicolls, "Locating facial features with an extended active shape model," in *Computer Vision*. Springer, 2008.
[47] S. Zhu, C. Li, C. C. Loy, and X. Tang, "Face alignment by coarse-to-fine shape searching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 4998–5006.
[48] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1. Dec. 2001, pp. I-511–I-518.
[49] R. Lienhart and J. Maydt, "An extended set of Haar-like features for rapid object detection," in *Proc. Int. Conf. Image Process.*, vol. 1. 2002, pp. I-900–I-903.

Shaoqing Ren received the B.S. degree from the University of Science and Technology of China, in 2011. He is currently pursuing the Ph.D. degree in a joint Ph.D. program between the University of Science and Technology of China and Microsoft Research Asia. His research interests include computer vision, especially detection and localization of general object and face.

Xudong Cao received the B.S. degree from Tsinghua University, in 2008. He joined Microsoft Research as an Associate Researcher in 2011. He is also interested in computer vision and machine learning. His current major research interests include face alignment and recognition.

Yichen Wei received the B.S. degree from the Computer Science Department, Peking University, in 2001, and the Ph.D. degree from the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, in 2006. He joined Microsoft Research in 2006. His research fields are in computer vision, especially in 3D reconstruction, object recognition, detection, and tracking.



Jian Sun received the B.S., M.S., and Ph.D. degrees from Xian Jiaotong University, in 1997, 2000, and 2003, respectively. He joined Microsoft Research Asia in 2003. He is currently a Principal Researcher with Microsoft Research. He is also interested in stereo matching and computational photography. His current two major research interests are interactive computer vision (user interface and vision) and Internet computer vision (large image collection and vision). He received the best paper award at the IEEE Conference on Computer Vision and Pattern Recognition in 2009.