Joint Cascade Face Detection and Alignment

Dong Chen[1], Shaoqing Ren[1], Yichen Wei[2], Xudong Cao[2], and Jian Sun[2]

[1] University of Science and Technology of China
{chendong,sqren}@mail.ustc.edu.cn
[2] Microsoft Research
{yichenw,xudongca,jiansun}@microsoft.com

Abstract. We present a new state-of-the-art approach for face detection. The key idea is to combine face alignment with detection, observing that aligned face shapes provide better features for face classification. To make this combination more effective, our approach learns the two tasks jointly in the same cascade framework, by exploiting recent advances in face alignment. Such joint learning greatly enhances the capability of cascade detection and still retains its realtime performance. Extensive experiments show that our approach achieves the best accuracy on challenging datasets, where all existing solutions are either inaccurate or too slow.

1 Introduction

Face detection is one of the mostly studied problems in vision [31]. The seminal work of Viola and Jones [26] has established the two foundation principles for practical solutions: 1) boosted cascade structure; 2) simple features. Most (if not all) realtime face detectors in academia and industry nowadays are based on the two principles. Such detectors work well for near-frontal faces under normal conditions but become less effective for faces non-frontal or under more wild conditions (lighting, expression, occlusion), because the simple features like Harr in the cascade training are insufficient to capture the more complex face variations.

Many works are on multi-view face detection [10,17,27,7]. They adopt a similar divide and conquer strategy: different face detectors are trained separately under different viewpoints or head poses, which are roughly quantized and estimated simultaneously. Because the viewpoint estimation problem is difficult as well and quantization also introduces inaccuracy, such training is more difficult and resulting detectors are usually slower or not accurate enough.

Recently, several new approaches that do not use boosted cascade have been proposed. Zhu et al. [32] used a mixture of deformable part models to capture large face variations under different viewpoints and expressions. The model is comprehensive and allows to estimate the head pose and facial points at the same time in addition to detection. Shen et al. [24] proposed the first exemplar-based face detector and exploited advanced image retrieval techniques to avoid the expensive sliding window search. Both methods are better than Viola-Jones style detectors on wild and challenging datasets [32,9]. However, those accurate

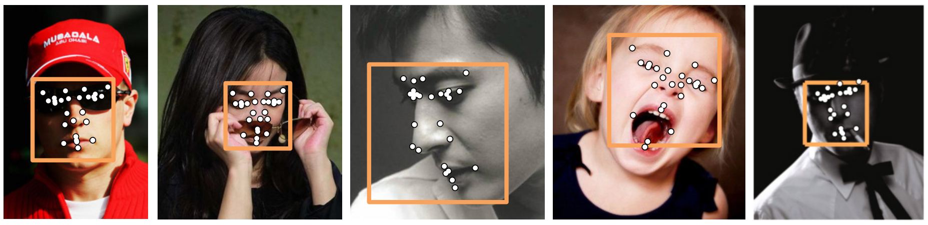

Fig. 1. Face detection and alignment results on challenging examples

detectors, as well as the state-of-the-art commercial face detector in Google Picasa, are all quite slow due to their high complexity. Detection in an image takes a few seconds and this makes such detectors unpractical for many scenarios.

In this work, we present a face detector that establishes the new state-of-the-art in terms of both accuracy and speed. It follows the "boosted cascade structure + simple features" principles. We use simple pixel differences as feature which bring advantages on the efficiency. Our detector takes only 28.6 milliseconds for a VGA image, more than 1000 times faster than [32]. It also achieves the best accuracy on the challenging datasets [32,9,22], significantly outperforms all existing academia solutions including [32,24], and is on bar with the commercial system in Google Picasa. Figure 1 shows our example detection results under large viewpoints, occlusion and poor lighting.

Our work is motivated by the observation that accurate face alignment (location of facial points) is helpful to distinguish faces/non-faces. In Section 2, we experimentally verified that a simple SVM classifier using facial point based features in the post processing can significantly improve the accuracy of a Viola-Jones detector. This finding is not surprising because face alignment finds corresponding parts between faces, makes them directly comparable, and simplifies the face/non-face classification problem. Similar observations have also been made in [32,14]. While the observation is clearly useful, the real problem is how to use it effectively. Previous methods [32,14] are too slow. Our post classification is also insufficient, because when high recall is expected, a cascade detector would return too many false positives and the SVM classifier would be slow too.

Our approach benefits from the recent advances in cascade face alignment [19,4,28,29,21]. In such works, the face shape is progressively updated via boosted regression. The regression learning in each stage not only depends on the image, but also depends on the estimated shape from the previous stage. Features learnt this way are called *shape indexed features*. Such features present more invariance to the geometric variations in the face shapes and they are crucial for high alignment accuracy and speed. As the cascade structure has been proven effective in detection and alignment, we propose to combine the two to benefit each other. In Section 3, we present a general cascade framework that unifies the two tasks. The detection learning is made more effective with embedded alignment information, and alignment is achieved simultaneously without performing it separately. In Section 4, we extend the recent state-of-the-art alignment method [21] under the new framework. We show how to use the same simple shape indexed features

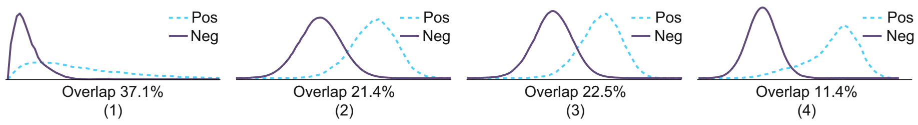

Fig. 2. The distribution of classification scores. (1): original cascade detector; (2)(3)(4) post SVM classifiers using three types of features, as described in Section 2.

also for detection so both are efficient. This results in new training and testing algorithms for joint cascade detection and alignment. In summary, we present the first joint cascade face detection and alignment method. We firstly show that simple shape indexed features are effective for detection as well. Extensive experiment results in Section 5 verify the superior performance of our approach in both accuracy and speed.

2 Alignment Helps Detection: A Post Classifier

We illustrates the effectiveness of using face alignment for detection on the challenging FDDB dataset [9]. We use the Viola-Jones detector in OpenCV with a low threshold to ensure a high recall. The detector outputs a lot of image windows, of which many are false positives. We split all the images into two parts, use the positive and negative output windows in the first part to train a linear SVM classifier, and test all the output windows in the second part. We call it a *post classifier* as it operates on the output of a cascade detector. All windows are resized into 96*96 pixels. We compare three types of features, without and with alignment.

1. we divide the window into 6*6 non-overlapping cells and extract a SIFT descriptor in each cell.
2. we use a fixed mean face shape with 27 facial points and extract a SIFT descriptor centered on each point.
3. we align the 27 facial points using the alignment algorithm in [21] and extract a SIFT descriptor centered on each point.

In the three cases, all the SIFT descriptors are concatenated as the final feature vector for the SVM. We plot different classification score distributions over the testing windows in Figure 2: (1) classification scores from original cascade detector; (2)(3)(4) SVM classification score of three types of features described above. It clearly shows that features at the aligned facial points are much more effective to separate those difficult windows.

While effective, the face alignment based post classifier is brute force and too slow for a standard cascade detector when a high recall is desired. In our experiment, a low threshold is set to achieve 99% recall of the OpenCV detector and each image outputs 3000 windows on average. Applying the post classifier for those windows takes a few seconds.

Table 1. Notations in this paper

category	notation	meaning
scalars	f	classification score;
	θ	bias threshold in classification;
	$y \in \{1, -1\}$	classification label;
vectors	$\mathbf{x}$	image window;
	$\mathbf{S}(\Delta\mathbf{S})$	face shape (increment);
functions	$\mathcal{C}$	a general weak classifier;
	$\mathcal{R}$	regression tree in [21];
	$\mathcal{CR}$	classification/regression tree in this work;
parameters and values	$L = 27$	# facial points;
	$T = 5$	# stages of cascade alignment;
	$N = 5400$	# total weak classifiers of cascade detection;
	$K = N/T = 1080$	# weak classifier/regressor in each stage;

3 A Unified Framework for Cascade Face Detection and Alignment

To better exploit the alignment information, we propose a unified framework for cascade face detection and alignment. We explain the notations when they are firstly used. They are also summarized in Table 1 for clarity and reference.

Cascade Detection. Without loss of generality, the classification score in the cascade detection can be written as

$$f^N = \sum_{i=1}^{N} \mathcal{C}^i(\mathbf{x}). \tag{1}$$

Each $\mathcal{C}^i$ is a weak classifier. To test an image window $\mathbf{x}$, the weak classifiers are sequentially evaluated and the window is rejected immediately whenever $f^n < \theta^n$ for any $n = 1, 2, ..., N$, where θ^n is the bias threshold. Therefore, the cascade detection is very fast because most negative image windows are rejected after evaluating only a few weak classifiers.

Cascade Alignment. Recent work [19] proposes a pose regression framework that combines pose indexed features with boosted regression. The framework has been shown highly effective for face alignment [4,28,29,21].

Let a face shape $\mathbf{S}$ be a $2L$ dimensional vector for L facial points. In the cascade regression, it is progressively updated through T stages as

$$\mathbf{S}^t = \mathbf{S}^{t-1} + \mathcal{R}^t(\mathbf{x}, \mathbf{S}^{t-1}), t = 1, ..., T. \tag{2}$$

Algorithm 1. The general testing algorithm for cascade face detection and alignment for an image window $\mathbf{x}$.

```
 1: initialize the face shape $\mathbf{S}$ as the mean shape in window of $\mathbf{x}$
 2: initialize the detection score $f = 0$
 3: for $t = 1$ to $T$ do
 4:     for $k = 1$ to $K$ do
 5:         $f = f + \mathcal{C}_k^t(\mathbf{x}, \mathbf{S})$
 6:         if $f < \theta_k^t$ then
 7:             return "not a face"
 8:         end if
 9:     end for
10:     $\Delta\mathbf{S} = \mathcal{R}^t(\mathbf{x}, \mathbf{S})$
11:     $\mathbf{S} = \mathbf{S} + \Delta\mathbf{S}$
12: end for
13: return "is a face with shape $\mathbf{S}$"
```

Each $\mathcal{R}^t$ is a regression function. It adds an increment to the estimated shape from the previous stage $\mathbf{S}^{t-1}$. It is learnt to minimize the shape residual error between the ground truth shape $\hat{\mathbf{S}}$ and estimated shape in the current stage, as

$$\mathcal{R}^t = \arg\min_{\mathcal{R}} \sum_i ||\hat{\mathbf{S}}_i - (\mathbf{S}_i^{t-1} + \mathcal{R}(\mathbf{x}_i, \mathbf{S}_i^{t-1}))||^2, \tag{3}$$

where index i iterates over all the training samples.

A Unified Framework. A key innovation in the cascade alignment framework is that each regressor $\mathcal{R}^t$ depends on the previous shape $\mathbf{S}^{t-1}$. During learning, the features are defined relative to $\mathbf{S}^{t-1}$, so called *pose/shape indexed features* [19,4]. Such features present better geometric invariance to the shape variations and they are crucial for the cascade framework.

We propose to apply such features in detection as well, by making the learning of weak classifier $\mathcal{C}^i(\mathbf{x})$ in Eq. (1) also dependent on the face shape. Note that the number of weak classifiers N in detection is usually hundreds or thousands. It is much larger than the number of stages T in alignment, which is usually less than 10[1]. To unify the learning of two tasks, we divide the N weak classifiers into the T stages. Each stage has $K = N/T$ weak classifiers and they depend on the face shape from the previous stage. The classification score in Eq. (1) is rewritten as

$$f = \sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{C}_k^t(\mathbf{x}, \mathbf{S}^{t-1}). \tag{4}$$

In principle, the regression and classification functions in the same stage t do not have to be learnt and applied simultaneously. Algorithm 1 illustrates a

[1] Using a large number of stages would cause the shape indexed features unstable and lead to poorer performance [4].

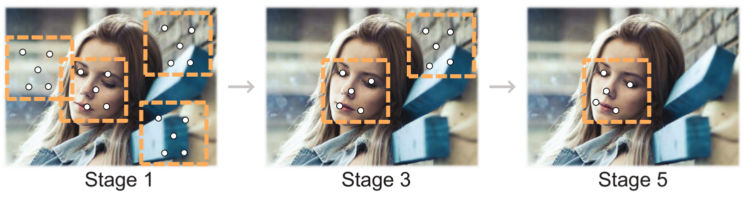

Fig. 3. Illustration of our cascade face detection and alignment

general testing algorithm for cascade detection and alignment, where the two parts are separate. However, in practice, we would expect that the two parts are related or even share the same features, because this is more efficient for both training and testing. Figure 3 illustrates the testing algorithm of our cascade face detection and alignment. The negative image windows are gradually rejected and the facial points of positive windows are gradually aligned stage by stage. In the next section, we present an effective joint learning approach.

4 Our Approach

As shown in Section 3, any cascade shape indexed face alignment method [4,28,29,21] can actually be used for cascade detection. We adopt the most recent and state-of-the-art approach [21], because it is most accurate, fastest, and easy to integrate the weak classifier learning for detection. Notably, it is significantly faster (dozens of times) than the previous best methods [4,28] and runs in thousands of frames per second to align dozens of facial points. Such high speed is crucial for a real time face detector as well.

We briefly review the work in [21] in Section 4.1 and describe how to extend it for joint alignment and detection in Section 4.2.

4.1 Review of Face Alignment in [21]

Its regression function $\mathcal{R}^t$ is simply the sum of K tree based regressors,

$$\mathcal{R}^t(\mathbf{x}, \mathbf{S}^{t-1}) = \sum_{k=1}^{K} \mathcal{R}_k^t(\mathbf{x}, \mathbf{S}^{t-1}). \tag{5}$$

Each $\mathcal{R}_k^t$ is a decision tree that stores a shape increment in every leaf. For the face window $\mathbf{x}$, it outputs the increment of the leaf into which $\mathbf{x}$ falls.

Learning of all $\mathcal{R}_k^t$ consists of two steps:

1. *Local learning of the tree structure*: for each of the L facial point, a standard regression forest [2] is learnt to estimate the increment of this point, using the shape indexed pixel difference features [4].
2. *Global learning of the tree output*: the point increments in the leaves are discarded. Instead, each leaf stores a shape increment and all such shape increments are optimized simultaneously by solving Eq. (3). Note that this is simply a global linear regression problem.

Algorithm 2. Our testing algorithm for cascade face detection and alignment for an image window $\mathbf{x}$. The model consists of all weak learners $\{\mathcal{CR}_k^t\}$ and classification thresholds $\{\theta_k^t\}$.

```
 1: initialize the face shape $\mathbf{S}$ as the mean shape in window of $\mathbf{x}$
 2: initialize the detection score $f = 0$
 3: for $t = 1$ to $T$ do
 4:     $\Delta\mathbf{S} = \mathbf{0}$
 5:     for $k = 1$ to $K$ do
 6:         $(f', \Delta\mathbf{S}') = \mathcal{CR}_k^t(\mathbf{x}, \mathbf{S})$
 7:         $f = f + f'$
 8:         if $f < \theta_k^t$ then
 9:             return "not a face"
10:         end if
11:         $\Delta\mathbf{S} = \Delta\mathbf{S} + \Delta\mathbf{S}'$
12:     end for
13:     $\mathbf{S} = \mathbf{S} + \Delta\mathbf{S}$
14: end for
15: return "is a face with shape $\mathbf{S}$"
```

The strength of this approach lies in the two step learning. The local learning focuses on an easier problem (one point regression in a local patch) and is more resistant to the noises in the simple pixel features, compared to [4]. Its learnt pixel features are more effective than the hand crafted SIFT features as in [28]. The global step enforces the dependence between individual facial points and reduces the local estimation errors. Given the fixed tree structures, this step achieves the global optimal solution. Therefore, the two step learning achieves a very strong local optimal solution in terms of Eq. (3).

4.2 Joint Learning of Detection and Alignment

Noticing that both the weak classifiers in Eq. (4) and tree regressors in Eq. (5) share a similar additive form, we propose to learn both classification and regression in a single decision tree. That is, each regression tree $\mathcal{R}_k^t$ in Eq. (5) is upgraded to a mixed tree $\mathcal{CR}_k^t$ that also outputs a classification score, in addition to its shape increment. Consequently, during testing, the classification and regression parts are evaluated simultaneously, as illustrated in Algorithm 2. As $\mathcal{CR}_k^t$ uses the same features for both classification and regression, such testing is faster than the general testing in Algorithm 1.

To learn a mixed classification/regression decision tree, we use a similar strategy as in hough forest [6]: in the split test of each internal node, we randomly choose to either minimize the binary entropy for classification (with probability ρ) or the variance of the facial point increments for regression (with probability $1 - \rho$). Intuitively, the parameter ρ should be larger in earlier stages to favor classification learning and reject easy negative samples more quickly. It should be smaller in later stages to favor regression learning and improve the alignment

Algorithm 3. Training of cascade and joint face detection and alignment.

```
1: Input: all training samples $\{\mathbf{x}_i\}$, class labels $\{y_i\}$
2: Input: ground truth shapes $\hat{\mathbf{S}}_i$ for positive samples, $y_i = 1$
3: Output: all weak learners $\{\mathcal{CR}_k^t\}$, classification thresholds $\{\theta_k^t\}$
4: set the initial face shapes $\mathbf{S}_i^0$ as random perturbations of the mean shapes in win-
   dows of $\mathbf{x}_i$
5: set all initial classification scores $f_i = 0$
6: for $t = 1$ to $T$ do
7:    for $k = 1$ to $K$ do
8:       for each training sample $i$ do
9:          compute its weight $w_i$ according to Eq. (6)
10:      end for
11:      select a point ($k$ mod $L$) for regression /*local learning in Section 4.1*/
12:      learn the structure of classification/regression tree $\mathcal{CR}_k^t$ as in Section 4.2
13:      for each tree leaf do
14:         set its classification score according to Eq. (7)
15:      end for
16:      for each training sample $i$ do
17:         update its classification score as $f_i = f_i + \mathcal{CR}_k^t(\mathbf{x}_i, \mathbf{S}_i^{t-1})$
18:      end for
19:      use all $\{f_i\}$ to set the bias $\theta_k^t$, according to a preset precision-recall condition

20:      remove samples whose $f_i < \theta_k^t$ from training set
21:      perform hard negative sample mining if negative samples are insufficient
22:   end for
23:   learn the shape increments of all leaves /* global learning in Section 4.1 */
24:   compute $\mathbf{S}_i^t$ for all samples according to Eq. (2) and (5)
25: end for
```

accuracy. We empirically make this parameter linearly decreasing with respect to the regression stage number t, that is, $\rho(t) = 1 - 0.1t, t = 1, ..., T$.

During split test for an internal node, we extend the shape indexed pixel difference features in [4] to multi-scale. Specifically, we generate three scales of images by down sampling the input image to half and one fourth. To generate a feature, we randomly choose an image scale, pick up two random facial points in the current shape, generate two random offsets with respect to the points and take the difference of the two offsetted pixels as the feature. We found the multi-scale pixel difference feature is more robust to noises and necessary for detection learning.

We use the RealBoost algorithm for the cascade classification learning. Before learning a decision tree, each sample i is associated with a weight w_i as

$$w_i = e^{-y_i f_i}, \tag{6}$$

where $y_i \in \{-1, 1\}$ is the face/non-face label and f_i is the current classification score. Such weights are used to compute a weighted binary entropy during the split test of internal node, as mentioned earlier.

In each tree leaf node, the classification score is computed as

$$\frac{1}{2}\ln\left(\frac{\sum_{\{i\in leaf\bigcap y_i=1\}} w_i}{\sum_{\{i\in leaf\bigcap y_i=-1\}} w_i}\right), \tag{7}$$

where the enumerator and denominator are the sum of positive and negative samples' weight in the leaf node, respectively.

The complete training algorithm of our cascade face detection and alignment is summarized in Algorithm 3. Note that all face shapes are normalized with respect to the windows containing them.

5 Experiments

We collected about $20,000$ face images and $20,000$ natural scene images without faces from web. All faces are manually labeled with 27 facial points. We use these images and their flipped versions for training. Only grayscale images are used. Most training parameters are listed in Table 1. For each classification/regression tree, the tree depth is set to 4 and number of split tests for each internal node is 2000. The training takes three days on a 16-core machine. Afterwards, we also train an alignment based SVM post classifier using the output windows in the training images, as described in Section 2. In detection we use standard sliding window search. Each window returned by the detector is then passed to the post classifier. All passing windows go through a non-maximum suppression process: the window rectangles are clustered and one window is selected in each cluster as the final detection.

For evaluation we use three challenging public datasets: FDDB [9], AFW [32] and CMU-MIT [22]. They do not include our training images. FDDB and AFW datasets are collected under wild conditions. They are widely used to evaluate the face detection methods [32,14,24]. CMU-MIT dataset is slightly out-of-date but its faces are low quality and quite different from those in FDDB, AFW and our training images. Thus, it is also challenging. We use it to test the generalization capability of our detector.

5.1 Effect of Alignment for Cascade Detection

The effect of alignment is verified by comparing our detector with a baseline detector without the alignment part. The baseline detector is trained in almost exactly the same way as ours. The only difference is that, line 24 in Algorithm 3 is ignored and the face shapes always remain as the initial mean shapes. It is essentially a standard Viola-Jones style cascade detector.

We compared the two detectors on FDDB dataset. We adjust the thresholds so that both have the same recalls and they can be fairly compared. Note that the post classifier is not used in this comparison. We firstly compare the false positive rates with respect to the number of tested weak classifiers. The left result of Figure 4 shows that our detector has much lower false positive rate

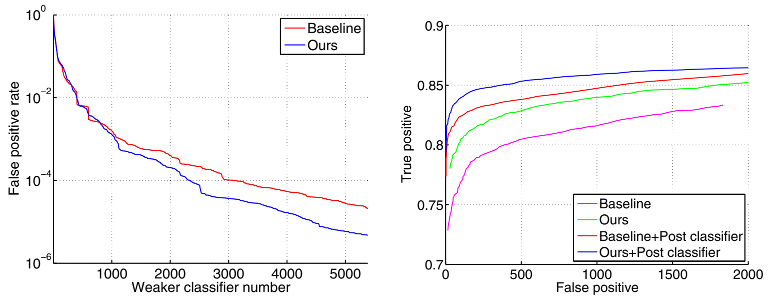

Fig. 4. Comparison of our detector and a baseline detector without alignment. Left: false positive rates all over the detection cascades. Right: recall versus number of false positives, with and without a post classifier.

all over the cascades and is more accurate. On average, each negative window requires 35.4 weak classifiers to reject it in the baseline detector, and this number in our detector is 26.1, indicating that our detector is also much faster. It demonstrates that the shape indexed features are more effective to distinguish negative samples, especially those hard ones in late stages.

The number of positive detection windows of the two detectors are close (22.2 for baseline and 21.5 for ours). However, the negative detections of the baseline is about 6 times larger than ours (82.1 and 13.3). Those hard negative samples cannot be easily removed by the post classifier. The comparison in the right of Figure 4 shows that although using a post classifier can significantly improve both detectors, the baseline detector is always worse than ours.

5.2 Comparison with the State-of-the-Art

In this part, our detector is comprehensively compared to the state-of-the-art. On FDDB dataset, we compare with all the published results in the platform [26,32,24,14,23,12,16,8,25,18,30,13,15]. It has two evaluation protocols: discrete and continuous. In the discrete setting, a detection window is considered correct if its “intersection-over-union” ratio with respect to an annotated face region is larger than 0.5. This criteria is commonly used in object detection evaluation. In the continuous setting, the overlapping ratio is used as a weight for every detection window. This criteria is much more strict. Figure 5 and 6 compare our approach with previous academia solutions and commercial systems, respectively. Our result outperforms all previous results by a large margin, under both protocols. Figure 9 shows our example detection results under various challenging conditions. Particularly, it is worth noting that that our detector already achieves a high recall (80.07%) when there are only 4 false positives, which are actually faces, as shown in Figure 9.

In AFW dataset, we use the precision-recall curves and the 50% overlapping criteria for evaluation, as in previous work [32,24]. We compare with the

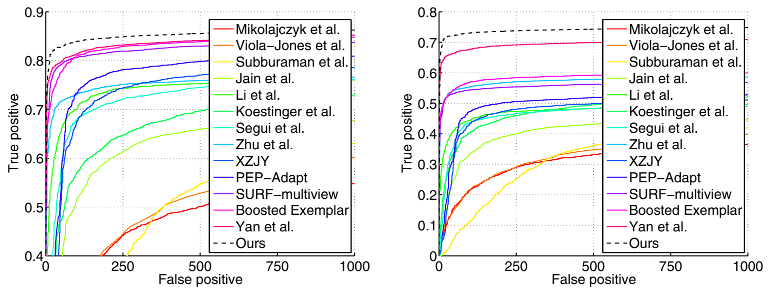

Fig. 5. Comparison with academia methods on FDDB dataset, under the discrete (left) and continuous (right) protocols

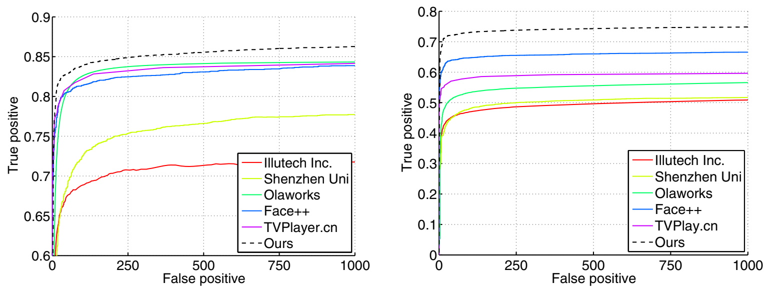

Fig. 6. Comparison with commercial systems on FDDB dataset, under the discrete (left) and continuous (right) protocols

following methods[2]: (1) OpenCV multi-view Viola-Jones; (2) Boosted multi-view face detector of [11]; (3) Deformable part model (DPM) [5]; (4) Mixture of trees [32]; (5) Face detection by retrieval [24]; (6) face.com's face detector; (7) Google Picasa's face detector. The results in the left of Figure 7 show that our detector is much better than previous academia methods and on bar with the best commercial face detector in Google Picasa.

In CMU-MIT dataset, we compare with following methods[3]: (1) Viola and Jones detector; (2) polygon-feature detector [20]; (3) recycling-cascade detector [3]; (4) soft cascade detector [1]; (5) SURF cascade [16]; (6) Google Picasa. The comparison results on CMU-MIT dataset is shown in the right of Figure 7. Our method is still the best. Note that the images in this dataset are quite different from others. Google Picasa only achieves 74% recall on this dataset.

5.3 Evaluation of Face Alignment

Our detector also outputs aligned face shapes as a by product. We evaluate the alignment accuracy on AFW dataset using the same settings in [32]. We compare

[2] The results of the other methods are provided by the author of [32,24].
[3] The results of the other methods are from [16].

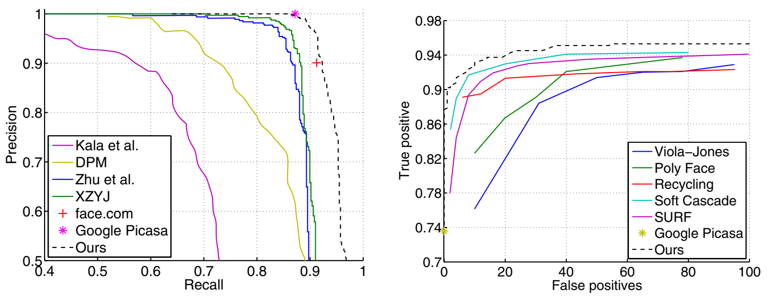

Fig. 7. Comparison with previous state-of-the-art methods on AFW (Left) and CMU-MIT (Right) dataset

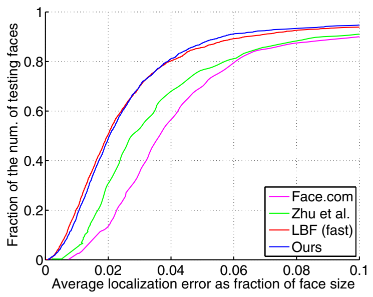

Fig. 8. Compare face alignment result with state-of-the-art methods

with the following methods: 1) face.com alignment system; 2) Zhu et al. [32]; 3) method in [21]. The results of first two methods are from [32]. The last is the state-of-the-art face alignment method. We implemented this method and used our detected faces as its input. Figure 8 shows that our face alignment accuracy is comparable to that in [21], not surprisingly, and is better than the first two.

5.4 Efficiency

Our detector is more efficient in terms of computation time and memory. We compare our detector with OpenCV detector and Zhu et al [32]. For all methods, we detect faces larger than 80×80 in a VGA image. Our detector takes 28.6 milliseconds using single thread on a 2.93 GHz CPU. This performance is more than 1000 times faster than Zhu et al. [32], which takes 33.8 seconds on the same image. Our detector approximates the speed of Viola-Jones detector in OpenCV, which is 23.0 milliseconds.

In the runtime, our detector needs only 15MB memory. Comparing with other methods, Li et al. [13] requires around 150MB memory and Shen et al. [24] requires 866MB memory[4], our detector is more practical for real scenarios such as mobile applications or on embedded devices.

[4] The results of the other methods are from [13].

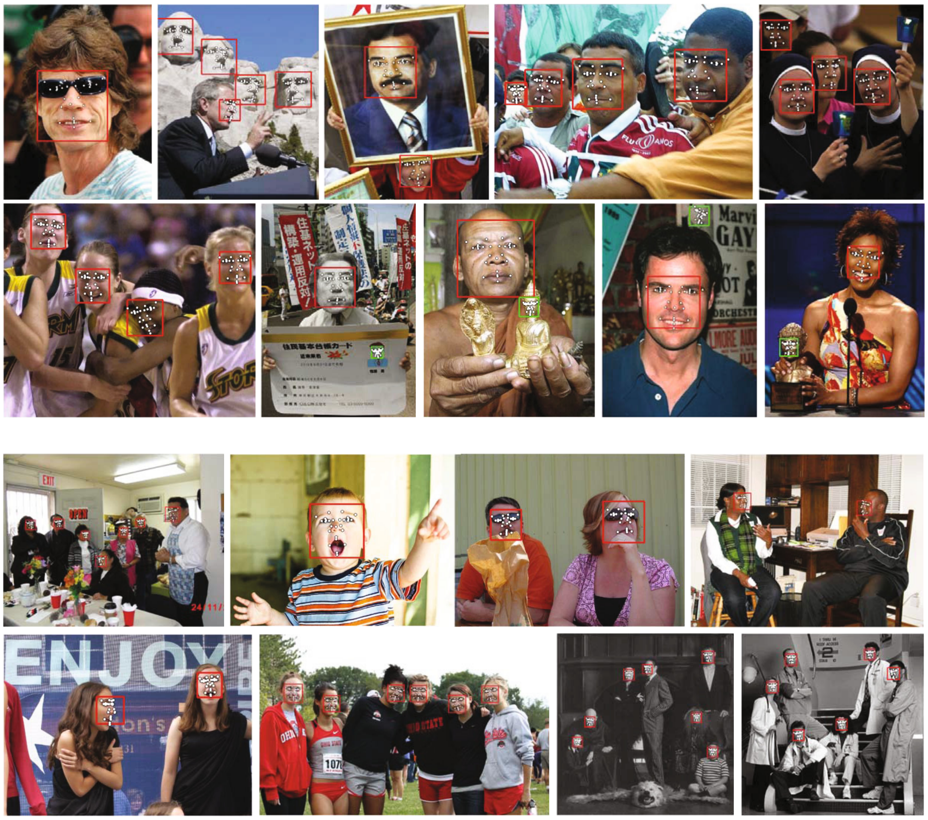

Fig. 9. The top two rows are our example detection results on FDDB dataset. Note that the last four images in the second row contain the four false positives in green rectangles, as described in the text. They are actually faces missed in the annotation. The last two rows are our example detection results on AFW and CMU-MIT datasets.

References

1. Bourdev, L.D., Brandt, J.: Robust Object Detection via Soft Cascade. In: Computer Vision and Pattern Recognition, vol. 2, pp. 236–243 (2005)
2. Breiman, L.: Random Forests. Machine Learning 45, 5–32 (2001)
3. Brubaker, S.C., Wu, J., Sun, J., Mullin, M.D., Rehg, J.M.: On the Design of Cascades of Boosted Ensembles for Face Detection. IJCV 77, 65–86 (2008)
4. Cao, X., Wei, Y., Wen, F., Sun, J.: Face Alignment by Explicit Shape Regression. In: Computer Vision and Pattern Recognition (2012)
5. Felzenszwalb, P.F., Girshick, R.B., McAllester, D.A., Ramanan, D.: Object Detection with Discriminatively Trained Part-Based Models. IEEE Transactions on Pattern Analysis and Machine Intelligence 32, 1627–1645 (2010)
6. Gall, J., Lempitsky, V.S.: Class-specific Hough forests for object detection. In: Computer Vision and Pattern Recognition, pp. 1022–1029 (2009)
7. Huang, C., Ai, H., Li, Y., Lao, S.: High-Performance Rotation Invariant Multiview Face Detection. IEEE Transactions on PAMI 29, 671–686 (2007)

8. Jain, V., Learned-Miller, E.: Online domain adaptation of a pre-trained cascade of classifiers. In: CVPR (2011)
9. Jain, V., Learned-Miller, E.: Fddb: A benchmark for face detection in unconstrained settings. Tech. Rep. UM-CS-2010-009, University of Massachusetts, Amherst (2010)
10. Jones, M.J., Viola, P.: Fast Multi-view Face Detection. In: CVPR (2003)
11. Kalal, Z., Matas, J., Mikolajczyk, K.: Weighted Sampling for Large-Scale Boosting. In: British Machine Vision Conference (2008)
12. Koestinger, M., Wohlhart, P., Roth, P.M., Bischof, H.: Robust face detection by simple means. In: DAGM 2012 CVAW Workshop
13. Li, H., Lin, Z., Brandt, J., Shen, X., Hua, G.: Efficient boosted exemplar-based face detection. In: CVPR (2014)
14. Li, H., Hua, G., Lin, Z., Brandt, J., Yang, J.: Probabilistic Elastic Part Model for Unsupervised Face Detector Adaptation. In: ICCV (2013)
15. Li, J., Zhang, Y.: Learning surf cascade for fast and accurate object detection. In: CVPR (2013)
16. Li, J., Wang, T., Zhang, Y.: Face detection using SURF cascade. In: International Conference on Computer Vision (2011)
17. Li, S.Z., Zhu, L., Zhang, Z., Blake, A., Zhang, H., Shum, H.: Statistical Learning of Multi-view Face Detection. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002, Part IV. LNCS, vol. 2353, pp. 67–81. Springer, Heidelberg (2002)
18. Mikolajczyk, K., Schmid, C., Zisserman, A.: Human detection based on a probabilistic assembly of robust part detectors. In: Pajdla, T., Matas, J(G.) (eds.) ECCV 2004. LNCS, vol. 3021, pp. 69–82. Springer, Heidelberg (2004)
19. Dollar, P., Welinder, P., Perona, P.: Cascaded pose regression. In: CVPR (2010)
20. Pham, M.T., Gao, Y., Hoang, V.D.D., Cham, T.J.: Fast polygonal integration and its application in extending haar-like features to improve object detection. In: Computer Vision and Pattern Recognition, pp. 942–949 (2010)
21. Ren, S., Cao, X., Wei, Y., Sun, J.: Face Alignment at 3000 FPS via Regressing Local Binary Features. In: Computer Vision and Pattern Recognition (2014)
22. Schneiderman, H., Kanade, T.: Probabilistic Modeling of Local Appearance and Spatial Relationships for Object Recognition. In: CVPR, pp. 45–51 (1998)
23. Segui, S., Drozdzal, M., Radeva, P., Vitri, J.: An integrated approach to contextual face detection. In: ICPRAM (2012)
24. Shen, X., Lin, Z., Brandt, J., Wu, Y.: Detecting and Aligning Faces by Image Retrieval. In: Computer Vision and Pattern Recognition (2013)
25. Venkatesh, B.S., Marcel, S.: Fast bounding box estimation based face detection. In: ECCV Workshop on Face Detection (2010)
26. Viola, P.A., Jones, M.J.: Rapid Object Detection using a Boosted Cascade of Simple Features. In: Computer Vision and Pattern Recognition, pp. 511–518 (2001)
27. Wu, B., Ai, H., Huang, C., Lao, S.: Fast Rotation Invariant Multi-View Face Detection Based on Real Adaboost. In: ICAFGR, pp. 79–84 (2004)
28. Xiong, X., DelaTorre, F.: Supervised Descent Method and its Applications to Face Alignment. In: Computer Vision and Pattern Recognition (2013)
29. Sun, Y., Wang, X., Tang, X.: Deep convolutional network cascade for facial point detection. In: Computer Vision and Pattern Recognition (2013)
30. Yan, J., Lei, Z., Wen, L., Li, S.Z.: The fastest deformable part model for object detection. In: CVPR (2014)
31. Zhang, C., Zhang, Z.: A Survey of Recent Advances in Face Detection (2010)
32. Zhu, X., Ramanan, D.: Face detection, pose estimation and landmark localization in the wild. In: Computer Vision and Pattern Recognition (2012)