Simultaneous Face Detection and Head Pose Estimation: A Fast and Unified Framework

Tingfeng Li and Xu Zhao(✉)

Department of Automation, Shanghai Jiao Tong University, Shanghai, China
{litingfeng,zhaoxu}@sjtu.edu.cn

Abstract. In this paper, we present a fast and unified framework for simultaneous face detection and 3D pose (pitch, yaw, roll) estimation of unconstrained faces using deep convolutional neural networks (CNN). Face detection is implemented with region-based framework as previous work like Faster RCNN. We model the pose estimation as a classification and regression problem: first divide continuous head poses into several discrete clusters, then adjust poses within each class with a class-specific regressor to achieve more accurate results. All classification and regressions for the two tasks are trained and tested simultaneously in one unified network. Our approach runs at 10 fps, which is the fastest implementation among the recently proposed methods as far as we know. Moreover, it is able to predict pose without using any 3D information. Extensive evaluations on several challenging benchmarks such as AFLW and AFW demonstrate the effectiveness of the proposed method with competitive results.

Keywords: Face detection · Head pose estimation · Convolutional neural networks

1 Introduction

For the past decades, cameras and smartphones have spread widely and countless photos are captured to record people's daily life. Making full use of these photos helps improve user experience, for example, users can look for photos taken with a particular friend [5]. Other Human Computer Interaction (HCI) devices like smart home requires devices to understand the expression of a person. And all the performances of other various face based applications, from face identification and verification to face clustering, tagging and retrieval, rely on accurate and efficient **face detection**. On the other hand, as another challenging task correlated with face analysis, **head pose estimation** has been found to be useful in human-robot interaction [14], driver attention detection [3] and social behavior analysis [24].

X. Zhao—This research is supported by the funding from NSFC programs (61673269, 61273285, U1764264).

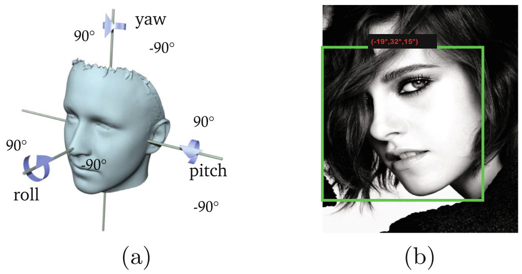


Fig. 1. Illustration of head pose estimation problem. (a) The pose of the head is described in the form of three rotation angles: pitch, yaw and roll [18]. (b) An example of predicted head pose.

These two tasks, face detection and head pose estimation, have traditionally been approached as independent problems, yet some methods solve them at the same time as separate components. As having common shared visual appearance of face regions, however, it will be more effective to solve them in a unified framework. And, pose angles could play as the latent variables and have immediate impact on visual appearance of face. It makes these two tasks closely correlated to each other and will be beneficial to model them in a unified framework. Recently, methods are proposed to solve the two tasks in a unified model [25,36,37]. However, these methods utilize facial landmarks to boost performance either explicitly or implicitly. In addition, multiple stages for training or testing makes their system not pure end-to-end, which discounts the final performance in the cost of time and accuracy.

In this paper, benefiting from deep Convolutional Neural Networks (CNN) architecture, our framework is designed to be more compact and efficient and completely end-to-end (see Fig. 2), that is, input an image, detect face and estimate head pose with only one network, without using face landmark information. In recent years, deep CNN have achieved significant performance improvement on benchmarks [5,23,26,34] for face detection. Furthermore, as a general deep CNN architecture for object detection, Ren *et al.* proposed an efficient region-based network Faster RCNN [27], showing outstanding performance in object detection. We adopt this architecture as our basic face detection network, then embed pose estimation module into this network, forming an end-to-end framework for both tasks. Specifically, we propose a novel implicit coarse-to-fine search scheme for pose estimation, which is embedded into the deep CNN architecture following multi-task learning pipeline.

For 3D head pose estimation, we expect the system to infer the orientation of person's head relative to camera coordinate, described by the rotation angles: pitch, yaw and roll, as shown in Fig. 1. Essentially, head pose estimation can be formulated as a regression problem, however, these three angles have to be recovered from monocular image to find the location of a pose in 3D pose space, precisely. Learning such a mapping function is challenging, as it is expected to be very sensitive to subtle change of face appearance, meanwhile, robust enough to noise interference. Besides, directly searching in the original space requires highly descriptive features and will be inefficient considering computational complexity. Hence we adopt a coarse-to-fine scheme, that is, we model head pose estimation as a process of discrete classification followed by fine-grained continuous regression. By firstly inferring the coarse pose through classification, a rough range of pose angles corresponding to a type of face appearance (e.g. frontal face) is determined. Then the pose is tuned again by a regressor with respect to this class to obtain a fine estimation. In so doing, we achieve state-of-the-art performance on challenging datasets with high computational efficiency.

To sum up, in this paper, we propose a fast and unified framework for accurate face detection and head pose estimation, and the main contributions can be recapped as follows.

- An effective unified framework is designed as an end-to-end network for both tasks: the input is an image containing faces, while the outputs are bounding boxes of faces and three orientation angles of each face, i.e., the head pose. The proposed method can run at 10 fps, which is fast enough for many real time applications.
- A novel pose-class-specific coarse-to-fine scheme is proposed to be embedded properly into the deep CNN network, for simultaneous pose estimation and face detection.
- New state-of-the-art performance is achieved on challenging unconstrained datasets such as AFLW and AFW.

2 Related Work

2.1 Face Detection

As a basic problem for face analysis, face detection has been studied for a long time. In numerous work, there exists two major categories, namely, rigid-templates based algorithm and Deformable Parts Model (DPM) based ones [35]. The early representative of the first category include the Viola-Jones face detection algorithm and its variations. DPM based model exploit abundant information on face, such as a potential deformation between facial parts, which enables them to combine face detection with facial part localization. Recent CNN based algorithms have shown exceptional results on general object detection, such architecture also have been investigated for face detection [5,23,26,34], in both categories.

In [26], DP2MFD based on Deformable Part Models [6] and deep pyramid features are proposed, reducing gap of DPM in training and testing on deep features. In [5], a single model is trained to fully capture faces in all orientations without pose or landmark annotation and shows good performance on popular benchmark datasets. In [34], facial parts responses by their spatial structure and arrangement are exploited. This method is very good at detecting faces under severe occlusion and variant poses.

In this work, we adopt the deep CNN framework for general object detection [27] as the basic architecture of face detection. Then we further extend it to an end-to-end system for simultaneous face detection and head pose estimation.

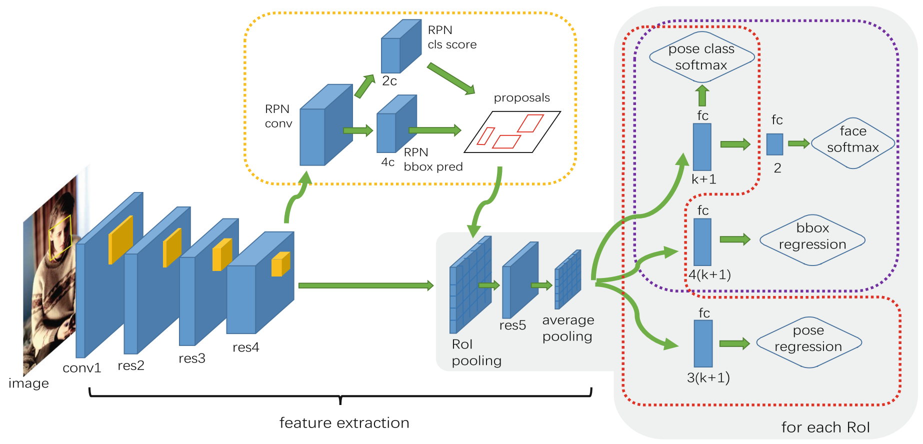


Fig. 2. The architecture of the proposed network. This network consists of four modules: feature extraction module (from conv1 layer to average pooling layer), RPN module (yellow dotted line), face detection module (purple dotted line, which is darker for white-black print), face pose estimation module (red dotted line). Best viewed in color. (Color figure online)

2.2 Head Pose Estimation

Head pose estimation aims at inferring the orientation of person's head relative to camera coordinate, which is very useful to many real life applications such as HCI and surveillance.

Several categories that describe the fundamental approaches underling its implementation have been used to estimate head pose [21]. We just briefly review some typical methods here. Appearance template methods [29] make comparison between a new head image with existing images labeled with discrete pose and find the most likely view. Detector array methods [13] train multiple detectors and assign a discrete pose to the detector with the greatest support. Nonlinear regression methods [22] use nonlinear regression tools to develop a functional mapping from the image or feature data to a head pose measurement. Manifold embedding methods [7] seek low-dimensional manifolds that model the continuous variation in head pose. New images can be embedded into these manifolds and then used for embedded template matching or regression. Recently, CNN-based model for head pose estimation had been developed, for example, in [20], deep CNN network is used to capture head pose in low-resolution RGB-D data.

Even more, it has been shown that learning correlated tasks simultaneously can boost the performance of individual tasks under the context of face detection and head pose estimation [2, 25, 36, 37]. The first work jointly addresses these two tasks is [36]. This method models each facial mark as a part then construct a mixture of trees to capture topological changes, much like DPM based model. Later on, this work was extended to [37]. Then Ranjan *et al.* proposed a multi-task learning framework, HyperFace [25], for face detection, landmark localization, pose estimation and gender recognition. Nevertheless, this work is not exactly end-to-end, since it requires a preprocess step such as Selective Search [31] to generate region of interests (RoIs). Counting all steps together it costs 3s per image in total. The most recent work KEPLER [16] is much faster and trained jointly, however, it is an iterative method requiring 5-stage training procedure to achieve adequate performance, which is very time consuming.

As all tasks work under deep CNN architectures, when compares to [16, 25], our network is purely end-to-end for both training and testing. Furthermore, as we adopt a novel pose-class-specific coarse-to-fine scheme, our method is computational efficient while maintaining high accuracy for both face detection and pose estimation.

3 Proposed Framework

The architecture of the proposed network is shown in Fig. 2. We devise the framework based on Faster RCNN [27], which is a region-based framework in general object detection. To further speed up and reduce computation cost, Region Proposal Network (RPN) is proposed to compute proposals that share convolutional layers with object detection networks. Following [27], we train both classifier and bounding box regressor with multi-task loss, which is convenient and has been demonstrated to facilitate object detection performance. In our method, pose-class-specific face detection and pose estimation is integrated naturally into this framework. A two-stage classification and bounding box regression solves face detection, where the first stage is to discriminate pose class defined by clustered angles and the second stage is to classify whether it is a face or not. As for pose estimation, the classification in the first stage helps predict rotation angles more accurately. In this section, we provide brief overview of the system and then describe each component in detail.

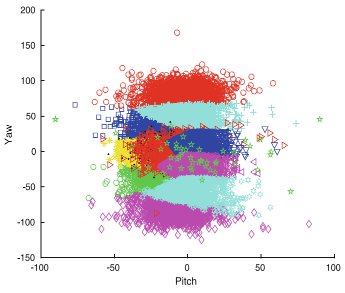
3.1 Network Architecture

Our network is constructed based on ResNet-50 [11]. Figure 2 presents the architecture of the whole framework. This framework consists of four major modules: feature extraction module, region proposal network (RPN) module, face detection module, and face pose estimation module. The first module is the backbone from input image to average pooling layer, and the rest three modules are denoted within dotted lines.

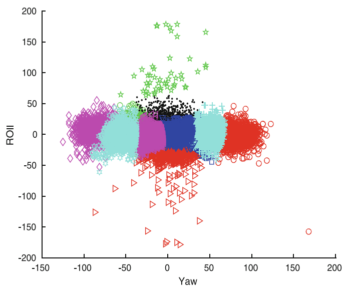
RPN Module. In the scenario of multi-category object detection, the RPN in Faster R-CNN [27] was developed as a class-agnostic detector. For single-category (face) detection, RPN is naturally a detector for the only category concerned. We specially tailor the RPN for face detection, as introduced in the following. We adopt the ResNet-50 pre-trained on the ImageNet dataset [15] as the backbone network rather than VGG-16 net [30]. The RPN is built on top of the $res4f$ layer, followed by an intermediate 3×3 convolutional layer with 256 channels and two sibling 1×1 convolutional layers for classification and bounding box regression (more details can be found in [27]). To deal with different scales and aspect ratios of objects, anchors were introduced in the RPN. An anchor is at each sliding location of the convolutional maps and thus at the center of each spatial window. Each anchor is associated with a scale and an aspect ratio. Following the default setting of [27], we use 3 scales (128^2, 256^2, and 512^2 pixels) and 3 aspect ratios (1:1, 1:2, and 2:1), leading to $c = 9$ anchors at each location. Therefore, for a convolutional feature map of size $W \times H$, we have at most $W \times H \times c$ possible proposals.

Face Detection Module. As in [27], this module is based on the Fast R-CNN detector [8] that uses the proposed regions. After feature map generated by $res4f$ layer, with the input of RoIs supplied by RPN, an RoI pooling layer is appended to extract a fixed-length feature vector in each RoI. Moreover, this layer helps to excavate local information. By taking full advantage of global and local information, the network is able to classify various scales of objects. After RoI pooling, features are extracted and down sampled by several $res5$ conv layers and the following average pooling layer respectively in every region. In the original ResNet, after the average pooling, a *fc* layer of 1000 output is added to classify 1000 categories in ImageNet. In our implementation, this last layer is removed to adapt our two tasks. Specifically, we replace it with three sibling *fc* layers: the first for two-stage classification, the second for bounding box regression and the third for pose regression. The two-stage classification is inspired by SubCNN, in which an additional *subcategory fc* layer is inserted before the *fc* layer for object class classification. In our network, pose class can be interpreted as subcategory, and the corresponding *fc* layer provides pose-class-specific feature for further face classification. For k pose classes, the corresponding *fc* layer outputs $k + 1$ dimensional vector with one additional dimension for the background class. As for the second sibling, we apply class-specific regression, that is, every regressor is tuned for each pose class. Finally, the detection module terminates at three output layers. A softmax function is applied at the first output layer directly based on the output of the "pose class fc" layer for pose class classification. The second output layer operates on the vector generated by "pose class fc" layer. And the last one outputs four real-valued numbers for each of the $k + 1$ pose classes. Each set of 4 values encodes refined bounding-box positions for one of the $k + 1$ classes.

Face Pose Estimation Module. As mentioned previously, estimation can be considered to have two steps: coarse estimation applied as classification and fine estimation implemented as regression. At first thought regression should come after classification. However, inspired by the design of prevalent class-specific bounding box regressors in object detection network architecture RCNN series [8,9,27], we are able to set these two steps as siblings leveraging the natural architecture of neural network. Specifically, we append a *fc* layer for the two parallel steps, i.e., classification and regression (see regions in red dotted line in Fig. 2). Note that we re-utilize the "pose class fc" layer with $k + 1$ outputs for pose classification. While "pose regression fc" layer has $3(k + 1)$ outputs, each



(a)



(b)

Fig. 3. Scatter plot of clusters projected. In this plot, only 9 poses have distinct location on this plane, while the remaining 3 clusters seems like can not be separated clearly. However, these pose classes can be differentiated easily when projected in roll direction.

set of 3 real values encodes the 3 refined rotation angles. For testing, we select the regressor corresponding to the highest pose class score to refine its pose in post process.

3.2 Pose Class

Subcategory has been widely utilized to facilitate object detection. Some methods discover subcategories by clustering objects according to the viewpoint [10]. In [32], Xiang *et al.* utilized subcategory to improve CNN-based detection, where clustering is preformed according to the orientation of the object for pedestrian and cyclist, and each cluster is considered to be a subcategory. Motivated by SubCNN, we introduce pose class for each face, which can be considered as subcategory in face detection. The pose class has two functions: first, it facilitates face detection. Second, it provides coarse pose information, narrowing down search space for further pose angle regression.

We apply kmeans [17] on AFLW dataset to form 12 clusters. Figure 3 shows an example scatter plot of clusters. In this plot, only 9 poses have distinct locations on Pitch-Yaw plane, while the remaining 3 clusters can not be separated clearly. However, when projected in roll direction, these 3 pose classes can be differentiated easily. Thus, on the whole, it is viable to divide continuous real valued pose angles into discrete classes.

3.3 Training

For RPN, we adopt loss function following [27]. While for the detection and estimation network, we utilize multi-task loss for joint pose class classification, face or non-face classification, bounding box regression and head pose regression (Eq. 1),

$$\begin{aligned} Loss = \lambda_1 L_{posecls}(p, k) + \lambda_2 L_{cls}(p', k') \\ + \lambda_3 [k \geqslant 1] L_{loc}(t^k, v) + \lambda_4 [k \geqslant 1] L_{pose}(a, a^*), \end{aligned} \quad (1)$$

where $\lambda_i, i \in \{1,2,3,4\}$ are loss weights to balance their contributions to the overall loss. Both classification losses are implemented with log loss: $L_{posecls}(p,k) = -\log p_k$, $L_{cls}(p',k') = -\log p'_{k'}$, where p is a probability distribution over $K+1$ pose classes, p' is a probability distribution over $K'+1$ classes, in our experiments, $K = 12, K' = 1$, k and k' are the truth pose label and the truth class label respectively. For bounding box regression, we use the *smooth* $L1$ loss.

$$L_{loc}(t^k, v) = \sum_{i \in \{x,y,w,h\}} smooth_{L_1}(t_i^k - v_i), \tag{2}$$

in which $t^k = (t_x^k, t_y^k, t_w^k, t_h^k)$ denotes a predicted bounding box for class k, which specifies the pixel coordinates of the center together with width and height of each proposal. And $v = (v_x, v_y, v_w, v_h)$ indicates true bounding box regression targets for pose class k. Each ground-truth bounding box $G = (G_x, G_y, G_w, G_h)$ is specified in the same way. Thus, target $v_i, i \in \{x,y,w,h\}$ is computed as follows,

$$v_x = (G_x - t_x^k)/t_w^k \tag{3}$$

$$v_y = (G_y - t_y^k)/t_h^k \tag{4}$$

$$v_w = \log(G_w/t_w^k) \tag{5}$$

$$v_h = \log(G_h/t_h^k) \tag{6}$$

The *smooth* $L1$ loss is defined as,

$$smooth_{L_1}(x) = \begin{cases} 0.5x^2 & if\ |x| < 1 \\ |x| - 0.5 & otherwise, \end{cases} \tag{7}$$

Similarly, the loss function for pose regression is defined as:

$$L_{pose}(a, a^*) = \sum_{i \in \{p,y,r\}} smooth_{L_1}(d_i - d_i^*), \tag{8}$$

where d_i^* and d_i denotes truth angles and estimated ones respectively. In back-propagation training, derivatives for the multi-task loss are back-propagated to the previous layers.

3.4 Testing

From a given test image, we first extract convolutional features. Then, RoIs are generated by the RPN module. As features in each RoI are pooled by RoI pooling layer from the last conv layer, the subsequent network is able to make predictions for all tasks. In post-process, we select top N boxes according to face softmax output. Then for each box, assign the correspongding pose class with the highest score out of k+1 pose class scores. After determining pose class, more accurate rotation angles are obtained by appling the pose-class-specific pose regression.

With the sharing of convolutional layers among RPN, detection and estimation networks, computation time is reduced substantially. In our experiments, we also found that even without sharing convolutional layers, the computation speed summed up by the two separate networks is still faster than [25] thanks to the RPN proposed by Ren *et al.* [27].

4 Experimental Validations

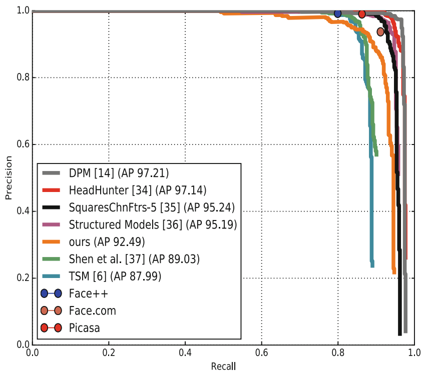
4.1 Datasets and Evaluation Metric

For face detection, we evaluate the proposed framework on the challenging AFW [36] dataset. For face pose estimation, we carry out experiments on both AFW and AFLW [18] datasets.

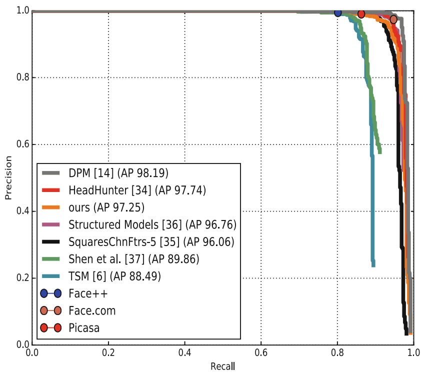
AFLW. To test the robustness of our approach for images from real senarios with challenging scale variations, cluttering background and significant shape changes, we utilize AFLW for training and testing to evaluate the estimation performance. AFLW contains 24386 faces annotated with pose (yaw, pitch and roll) and truth bounding box in 21997 real-world wild images. Head poses ranging from $0°$ to $120°$ for yaw and upto $90°$ for pitch and roll exhibiting a large variety in appearance (e.g., pose, expression, ethnicity, age, gender) as well as general imaging and environmental conditions. We use exactly the same 1000 images randomly selected by [25] for testing, all other images for training.

AFW is a very popular benchmark for evaluation of both face detection and head pose estimation algorithms. This dataset provides 205 images with 468 faces in the wild with yaw degree up to $90°$. The images tend to contain cluttered background with large variations in face viewpoint, illumination and appearance (aging, sunglasses, make-ups, skin color, expression etc.). Each face is labeled with a bounding box, and a discretized viewpoint ($-90°$ to $90°$ every $15°$) along pitch and yaw directions and (left, center, right) viewpoints along the roll direction. In consistent with other methods, we select 341 faces with height greater than 150 pixels following the protocol of [36].

Evaluation Metric. Following previous works, we use Average Precision (AP) [4], and precision-recall curves to test our face detector on AFW. We demonstrate our estimation results with mean error over all samples and Cumulative Error Distribution (CED) curve. This curve provides the fraction of faces with predicted pose within $\pm 15°$ error tolerance. Note that we only evaluate on discretized pose predictions rounded to the nearest $15°$ on AFW, e.g., if a predicted angle is $85°$, then it is rounded to $90°$ rather than $75°$ for evaluation.



(a)



(b)

Fig. 4. Precision-recall curves for face detection on AFW dataset. (a) evaluation IoU threshold 0.5, (b) evaluation IoU threshold 0.3. The numbers in the legend are the average precision for the corresponding IoU threshold.

4.2 Setting

The proposed approach is tested on Intel(R) Xeon(R) 2.10 GHz CPU with 32 GB RAM, NVIDIA TITAN X GPU. As in Faster RCNN, we use image-centric sampling, each SGD mini-batch is constructed from a single image. In RPN subnetwork, a mini-batch is expected to have 64 positive RoIs and 64 negative RoIs selected by random sampling for simplicity. For detection and estimation network, a mini-batch is constructed from a single image with 128 RoIs with 1:3 ratio of positive and negative samples. We use a learning rate of 0.001 in the beginning and drop 0.1 every 50k mini-batch iterations for all 200k mini-batches. We use a weight decay of 0.0005 and a momentum of 0.9 [15]. All the experiments including training and testing were performed using the Caffe [12] framework.

4.3 Face Detection

Face Detection Results and Error Analysis. Notice that we evaluate on different training and testing dataset. As pointed out by Mathias *et al.* [19], one important problem in the evaluation of face detection methods is the mismatch of face annotations in the training and testing stages. Specifically, it is not so obvious about how to define the rectangle around face for profile and semi-profile views. In AFLW, the rectangle tends to be larger and more like a square. When it comes to profile face, the rectangle extends to non-face areas. While in AFW, the rectangle is likely to be tighter.

To solve this problem, we follow the remedial method proposed by Mathias *et al.* [19] to search for a global rigid transformation of the detection outputs to maximize the overlapping ratio with the ground-truth annotations with the shared evaluation tool [19]. However, our annotations cannot be simply linearly

Table 1. AP comparison of our method with different configurations.

IoU 0.3	IoU 0.5	Transformation	AP (%)
√		√	**97.25**
	√	√	92.49
	√		81.55
√			96.85

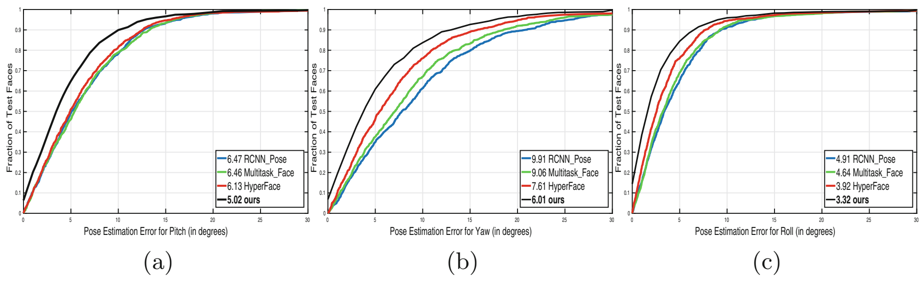


Fig. 5. Performance evaluation of pose estimation on AFLW dataset for (a) roll (b) pitch and (c) yaw angles respectively. The numbers in the legend are the mean error in degrees for the respective pose angles.

mapped to AFW annotations. After the global transformation step the mismatches still exist. According to our analysis, the inferior performance on AFW may primarily be caused by poor localization resulting from annotation mismatch. To demonstrate this, we set evaluation IoU to 0.3 and 0.5 on all methods to make a fair comparison. Table 1 presents results with different IoU thresholds with or without transformation. When IoU is decreased from 0.5 to 0.3, we can see that with transformation, our method is improved a lot from 92.49% to 97.25%, and accuracy is improved even more without transformation, from 81.55% to 96.85%. Moreover, even with consistent IoU threshold 0.3, AP of our method increases to 0.4 by applying transformation, suggesting that our method tends to predict results following the annotation style of AFLW dataset. We also provide precision-recall curves produced by different methods with transformation under IoU 0.5 and 0.3 in Fig. 4. We compare our approach with the following: (1) Deformable part model (DPM) [6], (2) HeadHunter [19], (3) SquaresChnFtrs-5 [1], (4) Structured Models [33], (5) shen *et al.* [28], (6) TSM [36], (7) Google Picasa‘s face detector, manually scored by inspection, (8) Face.com‘s face detector, (9) Face++‘s face detector. As can be seen, AP of our method increases largely while others only have slightly improvement. Therefore, we can conclude that the inferiority of our method comparing to others is mainly due to annotation mismatch between training and testing datasets.

Table 2. Comparison of head pose estimation with other methods on AFLW dataset, where speed is for both tasks, i.e., face detection and pose estimation. HyperFace is based on AlexNet, while HF-ResNet is based on ResNet-101. ours_vgg is based on VGG16, while ours_res is based on ResNet-50.

Methods	Mean absolute error			Speed
	Pitch	Yaw	Roll	
HyperFace [25]	6.13	7.61	3.92	0.33 fps
HF-ResNet [25]	5.33	6.24	**3.29**	0.33 fps
KEPLER [16]	5.85	6.45	8.75	3∼4 fps
ours_vgg	5.36	6.51	3.41	**10 fps**
ours_res	**5.02**	**6.01**	3.32	3 fps

4.4 Head Pose Estimation

AFLW. We compare MAE of different pose angles and speed with other state-of-the-arts in Table 2. As we can see, except for roll, the proposed method achieves the best performance with the speed of 3 fps on MAE when using ResNet-50. In addition, when we apply VGG16 as backbone, the method is accelerated to 10 fps while maintains relatively lower MAE in both pitch and roll comparing to HyperFace, only slightly higher than KEPLER in yaw. This indicates that our method is the best in both accuracy and efficiency. Figure 5 shows the cumulative error distribution curves on AFLW dataset in the order of pitch, yaw and roll respectively. It can be seen that the proposed method outperforms others with more accurate predictions. It is worth noticing that either in Table 2 or Fig. 5, the estimated yaw angle is less good as other two angles. One possible explanation is that the distribution of yaw in AFLW dataset is more decentralized. We will discuss it in detail in Subsect. 4.5.

Table 3. Comparison on AFW dataset.

Methods	Accuracy ($\leq 15°$)
Multi.HoG	74.6%
Multi.AAMs	36.8%
Face.com	64.3%
FaceDPL [37]	89.4%
KEPLER [16]	96.67%
HyperFace [25]	97.7%
HF-ResNet [25]	98.5%
Ours	98.24%

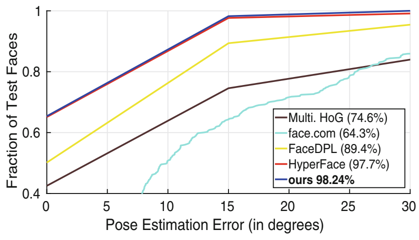


Fig. 6. Cumulative error distribution curves for pose estimation on AFW dataset.

Table 4. Mean average error comparison on AFLW, where nocls indicates no clustering applied, 12clsAll means clustering into 12 groups by three directions, and 12clsYaw denotes clustering into 12 groups by yaw.

Methods	Mean absolute rrror			AP
	Pitch	Yaw	Roll	
nocls	5.21	6.46	3.33	94.12%
12clsAll	5.13	6.16	**3.32**	94.62%
12clsYaw	**5.02**	**6.01**	**3.32**	**95.03%**

AFW. To demonstrate the capability of generalization of our method, we conduct experiments on AFW dataset. Because the ground-truth yaw angles are provided in multiples of 15°, we round-off our predicted yaw to the nearest 15° for evaluation in consistent with the previous works. In Table 3, we compare our method with the following: (1) Multiview AAMs: an AAM trained for each viewpoint, (2) face.com, (3) Multiview HoG, (4) FaceDPL [37], (5) HyperFace [25], (6) KEPLER [16]. The proposed method scores 98.24% when allowing ±15° error tolerance, achieving comparable performance. Note that HF-ResNet [25] uses ResNet-101, a deeper and better backbone, while our implementation is based on ResNet-50. This difference might potentially affects the final performance. Figure 6 shows cumulative error distribution curves of the proposed method as well as some of other methods. It is clear that the proposed algorithm achieves the state-of-the-art, and is able to predict yaw in the range of ±15° for more than 98% of the faces.

4.5 Does Pose Class Help?

To test whether pose class boost performance or not in our work, we conduct experiments with different settings. First we calculate the covariance of three angles on AFLW dataset,

$$cov_{pose} = \begin{pmatrix} 0.0547 & 0.0097 & 0.0002 \\ 0.0097 & 0.5324 & -0.0168 \\ 0.0002 & -0.0168 & 0.0601 \end{pmatrix}$$

The 3×3 matrix is the covariance matrix of angles in the order of pitch, yaw and roll. From the diagonal elements, we can see that the variances of pitch and roll are 0.0547 and 0.0601 respectively. It is very small comparing to that of yaw, which is 0.5324. Other elements indicate that the three angles are almost independent to each other. Therefore, yaw contributes most to the variance of 3D pose and clustering in this direction should be more effective. We verify this guess by making comparison between *no clustering*, *clustering only on the basis of yaw direction* and *of all directions*. When *no clustering* is applied, the *fc* layer before pose class softmax is removed, and *fc* layers before bbox regression and pose regression are set to 8 and 6 respectively. Table 4 shows the results of MAE

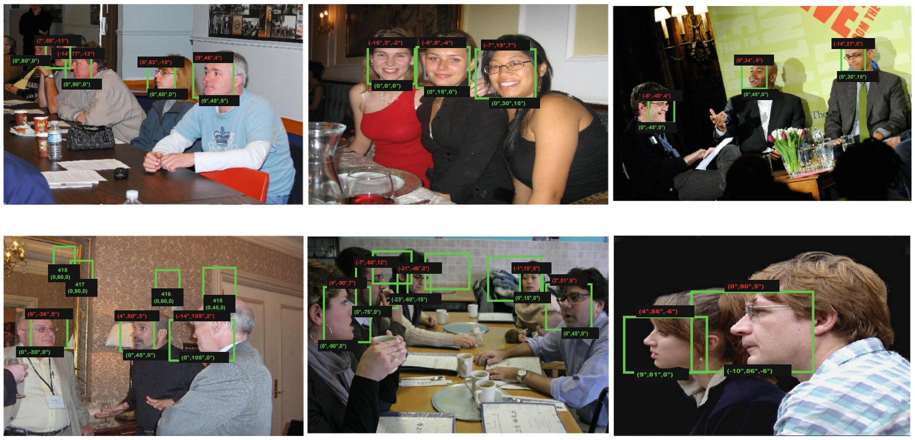


Fig. 7. Qualitative results of our method. Pose estimates for each face are shown on top of the boxes in the order (pitch, yaw, roll), groundtruth angles are shown at the bottom. Predictions within ±15° error tolerance could be considered as adequate results.

and AP on AFLW dataset. From this table, we observe that either on MAE or AP, clustering by yaw achieves the best and clustering by all directions performs better than no clustering, demonstrating that pose class indeed has impact on both pose estimation and face detection. Furthermore, generating pose classes along the principle direction significantly boosts the performance (Fig. 7).

5 Conclusions

In this paper, we propose an accurate and cost efficient framework for simultaneous face detection and 3D head pose estimation. The entire face detection module is a region-based method as Faster RCNN. While for head pose estimation, pose angles are firstly clustered into discrete groups. Then the regression problem becomes a classification problem. Finally a class-specific regressor is trained to refine the pose in each class. The estimation module is combined to detection module by sharing convolutional features in early layers. Extensive results on challenging unconstrained datasets demonstrates the effectiveness of our method.

References

1. Benenson, R., Mathias, M., Tuytelaars, T., Van Gool, L.: Seeking the strongest rigid detector. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3666–3673 (2013)
2. Chen, D., Ren, S., Wei, Y., Cao, X., Sun, J.: Joint cascade face detection and alignment. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8694, pp. 109–122. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10599-4_8

3. Doshi, A., Trivedi, M.M.: Head and eye gaze dynamics during visual attention shifts in complex environments. J. Vis. **12**(2), 9–9 (2012)
4. Everingham, M., Van Gool, L., Williams, C., Winn, J., Zisserman, A.: The pascal visual object classes challenge 2012 (voc2012) results (2012) (2010). http://www.pascal-network.org/challenges/VOC/voc2011/workshop/index.html
5. Farfade, S.S., Saberian, M.J., Li, L.J.: Multi-view face detection using deep convolutional neural networks. In: Proceedings of the 5th ACM on International Conference on Multimedia Retrieval, pp. 643–650. ACM (2015)
6. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE Trans. Pattern Anal. Mach. Intell. **32**(9), 1627–1645 (2010)
7. Fu, Y., Huang, T.S.: Graph embedded analysis for head pose estimation. In: 7th International Conference on Automatic Face and Gesture Recognition FGR 2006, p. 6. IEEE (2006)
8. Girshick, R.: Fast R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448 (2015)
9. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 580–587 (2014)
10. Gu, C., Ren, X.: Discriminative mixture-of-templates for viewpoint classification. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6315, pp. 408–421. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15555-0_30
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
12. Jia, Y., et al.: Caffe: convolutional architecture for fast feature embedding. In: Proceedings of the 22nd ACM International Conference on Multimedia, pp. 675–678. ACM (2014)
13. Jones, M., Viola, P.: Fast multi-view face detection. Mitsubishi Electric Res. Lab TR-20003-96 **3**(14), 2 (2003)
14. Katzenmaier, M., Stiefelhagen, R., Schultz, T.: Identifying the addressee in human-human-robot interactions based on head pose and speech. In: International Conference on Multimodal Interaction, pp. 144–151 (2004)
15. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
16. Kumar, A., Alavi, A., Chellappa, R.: Kepler: keypoint and pose estimation of unconstrained faces by learning efficient H-CNN regressors. In: 2017 12th IEEE International Conference on Automatic Face Gesture Recognition (FG 2017), pp. 258–265, May 2017. https://doi.org/10.1109/FG.2017.149
17. Lloyd, S.: Least squares quantization in PCM. IEEE Trans. Inf. Theor. **28**(2), 129–137 (1982)
18. Koestinger, M., Wohlhart, P., Roth, P.M., Bischof, H.: Annotated facial landmarks in the wild: a large-scale, real-world database for facial landmark localization. In: Proceedings of First IEEE International Workshop on Benchmarking Facial Image Analysis Technologies (2011)
19. Mathias, M., Benenson, R., Pedersoli, M., Van Gool, L.: Face detection without bells and whistles. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8692, pp. 720–735. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10593-2_47

20. Mukherjee, S.S., Robertson, N.M.: Deep head pose: gaze-direction estimation in multimodal video. IEEE Trans. Multimedia **17**(11), 2094–2107 (2015)
21. Murphy-Chutorian, E., Trivedi, M.M.: Head pose estimation in computer vision: a survey. IEEE Trans. Pattern Anal. Mach. Intell. **31**(4), 607–626 (2009)
22. Osadchy, M., Cun, Y.L., Miller, M.L.: Synergistic face detection and pose estimation with energy-based models. J. Mach. Learn. Res. **8**(May), 1197–1215 (2007)
23. Qin, H., Yan, J., Li, X., Hu, X.: Joint training of cascaded CNN for face detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3456–3465 (2016)
24. Ramanathan, S., Yan, Y., Staiano, J., Lanz, O., Sebe, N.: On the relationship between head pose, social attention and personality prediction for unstructured and dynamic group interactions. In: ICMI, pp. 3–10 (2013)
25. Ranjan, R., Patel, V.M., Chellappa, R.: Hyperface: a deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. In: IEEE Transactions on Pattern Analysis and Machine Intelligence, p. 1 (2017). https://doi.org/10.1109/TPAMI.2017.2781233
26. Ranjan, R., Patel, V.M., Chellappa, R.: A deep pyramid deformable part model for face detection. In: 2015 IEEE 7th International Conference on Biometrics Theory, Applications and Systems Biometrics Theory, Applications and Systems (BTAS), pp. 1–8. IEEE (2015)
27. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, pp. 91–99 (2015)
28. Shen, X., Lin, Z., Brandt, J., Wu, Y.: Detecting and aligning faces by image retrieval. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3460–3467 (2013)
29. Sherrah, J., Gong, S., Ong, E.J.: Face distributions in similarity space under varying head pose. Image Vis. Comput. **19**(12), 807–819 (2001)
30. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
31. Uijlings, J.R., Van De Sande, K.E., Gevers, T., Smeulders, A.W.: Selective search for object recognition. Int. J. Comput. Vis. **104**(2), 154–171 (2013)
32. Xiang, Y., Choi, W., Lin, Y., Savarese, S.: Subcategory-aware convolutional neural networks for object proposals and detection. arXiv preprint arXiv:1604.04693 (2016)
33. Yan, J., Zhang, X., Lei, Z., Li, S.Z.: Face detection by structural models. Image Vis. Comput. **32**(10), 790–799 (2014)
34. Yang, S., Luo, P., Loy, C.C., Tang, X.: From facial parts responses to face detection: a deep learning approach. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3676–3684 (2015)
35. Zafeiriou, S., Zhang, C., Zhang, Z.: A survey on face detection in the wild: past, present and future. Comput. Vis. Image Underst. **138**, 1–24 (2015)
36. Zhu, X., Ramanan, D.: Face detection, pose estimation, and landmark localization in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2879–2886. IEEE (2012)
37. Zhu, X., Ramanan, D.: FACEDPL: detection, pose estimation, and landmark localization in the wild. Preprint **1**(2), 6 (2015)