Detecting Face with Densely Connected Face Proposal Network

Shifeng Zhang, Xiangyu Zhu, Zhen Lei, Hailin Shi, Xiaobo Wang, Stan Z. Li

CBSR&NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing, China
University of Chinese Academy of Sciences, Beijing, China
{shifeng.zhang,xiangyu.zhu,zlei,hailin.shi,
xiaobo.wang,szli}@nlpr.ia.ac.cn

Abstract. Accuracy and efficiency are two conflicting challenges for face detection, since effective models tend to be computationally prohibitive. To address these two conflicting challenges, our core idea is to shrink the input image and focus on detecting small faces. Specifically, we propose a novel face detector, dubbed the name Densely Connected Face Proposal Network (DCFPN), with high performance as well as real-time speed on the CPU devices. On the one hand, we subtly design a lightweight-but-powerful fully convolutional network with the consideration of efficiency and accuracy. On the other hand, we use the dense anchor strategy and propose a fair L1 loss function to handle small faces well. As a consequence, our method can detect faces at 30 FPS on a single 2.60GHz CPU core and 250 FPS using a GPU for the VGA-resolution images. We achieve state-of-the-art performance on the AFW, PASCAL face and FDDB datasets.

Keywords: Face detection · Small face · Region proposal network

1 Introduction

Face detection is one of the fundamental problems in computer vision. With the great progress, face detection has been successfully applied in our daily life. However, there are still some tough challenges in the uncontrolled face detection problem. The challenges mainly come from two requirements for face detectors: 1) The large variation of facial changes requires face detectors to accurately address a complicated face and non-face classification problem; 2) The large search space of arbitrary face positions and sizes further imposes a time efficiency requirement. These two requirements are conflicting, since high-accuracy face detectors tend to be computationally expensive.

To meet these challenges, face detection has been studied mainly in two ways. One way is cascade based methods and it starts from the pioneering work [1]. After that, a number of improvements to the Viola-Jones face detector have been proposed in the past decade [2]. The other way is CNN [3] based methods and some works based on R-CNN [4] have demonstrated the state-of-the-art performance on face detection tasks.

However, these two ways focus on different aspects. The former tends to great efficiency while the latter cares more about high accuracy. To perform well on both speed and accuracy, one natural idea is to combine the advantages of them. Therefore, cascade CNN based methods [5] are proposed that put features learned by CNN into cascade framework so as to boost the performance and keep efficient. However, there are two problems in cascaded CNN based methods: 1) Their speed is negatively related to the number of faces on the image. The speed would dramatically degrade as the number of faces increases; 2) The cascade based detectors optimize each component separately, making the training process extremely complicated and the final model sub-optimal.

Therefore, it is still one of the remaining open issues for practical face detectors to achieve real-time speed on CPU as well as maintain high performance. In this paper, we develop a state-of-the-art face detector with CPU real-time speed. Our core idea is to **shrink the input image and focus on detecting small faces**. Reducing high-resolution images into low-resolution images can significantly improve the detection speed, but it also results in smaller faces that need to pay more attention. Specifically, our DCFPN has a lightweight-but-powerful network with the consideration of efficiency and accuracy. Besides, we use the dense anchor strategy [6] and propose a fair L1 loss to handle small faces well. As a consequence, for VGA images to detect faces bigger than 40 pixels, the DCFPN can run at 30 FPS on a single CPU core and 250 FPS on a GPU card. For clarity, the main contributions of this work can be summarized as four-fold:

- We develop a novel face detector with real-time speed on the CPU devices;
- We design a lightweight-but-powerful fully convolution network for face detection;
- We propose a fair L1 loss and use dense anchor strategy to handle small faces well;
- We achieve state-of-the-art performance on common face detection benchmarks.

2 Related work

Face detection approaches can be roughly divided into two different categories. One is based on hand-craft features, and the other one is built on CNN. This section briefly reviews them and refer more detailed survey to [2,7,8].

Hand-craft based methods. The milestone work of Viola-Jones [1] proposes to use Haar feature, Adaboost learning and cascade inference for face detection. After that, many subsequent works focus on new local features [9,10], new boosting algorithms [11,12,13] and new cascade structures [14,15,16]. Besides the cascade framework, the deformable part model (DPM) [17] is introduced into face detection task by [18,19,20,21,22] , which use supervised parts, more pose partition, better training or more efficient inference to achieve better performance.

CNN based methods. They show advantages in face detection recently. CCF [23] uses boosting on top of CNN features for face detection. Faceness [24] trains fully convolutional networks (FCN) to generate heat map of facial parts and then use the heat map to generate face proposals. CascadeCNN [5] uses six cascaded CNNs to efficiently reject backgrounds in three stages. STN [25] proposes a new Supervised Transformer Network and a ROI convolution for face detection. MTCNN [26] presents a multi-task cascaded CNNs based framework for joint face detection and alignment.

Generally, hand-craft based methods can achieve real-time speed on the CPU devices, but they are not accurate enough for the uncontrolled face detection problem. With learned feature and classifier directly from the image, CNN based methods can differentiate faces from highly cluttered backgrounds, while they are too time-consuming to reach real-time speed. Notably, our proposed DCFPN is able to achieve real-time speed on the CPU devices as well as maintain state-of-the-art detection performance.

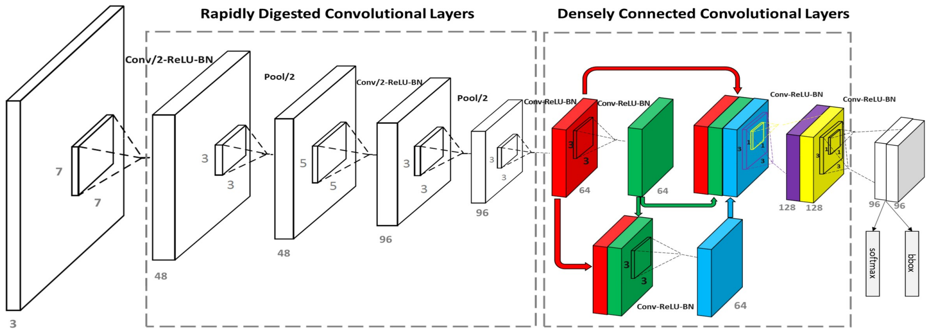


Fig. 1. The structure of DCFPN.

3 Densely connected face proposal network

This section presents detail of DCFPN. It includes three key components: the lightweight-but-powerful architecture, the dense anchor strategy and the fair L1 loss.

3.1 Lightweight-but-powerful architecture

The architecture of DCFPN encourages feature reuse and leads to a substantial reduction of parameters. As illustrated in Fig. 1, the whole architecture consists of two parts.

Rapidly Digested Convolutional Layers (RDCL). It is designed for high efficiency via quickly reducing the image spatial size by 16 times with narrow but large kernels. On one side, face detection is a two classification problem and does not require very wide network, hence the narrow kernels are powerful enough and can result in faster running speed, especially for CPU devices. On the other side, the large kernels are to alleviate the information loss brought by spatial size reducing.

Densely Connected Convolutional Layers (DCCL). Each layer in DCCL is directly connected to every other layer in a feed-forward fashion, and DCCL ends with two micro inception layers. There are two motivations. Firstly, the DCCL is designed to enrich the receptive field of the last convolutional layer that is used to predict the detection results. As listed in Tab. 1, the last convolutional layer of DCFPN has a large scope of receptive field from 75 to 235 pixels, which is consistent with our default anchors and is important for the network to learn visual patterns for different scales of faces. Secondly, the DCCL aims at combining coarse-to-fine information across deep CNN models to improve the recall rate and precision of detection. Since the information of the interest region is distributed over all levels of CNN with multiple level abstraction and they should be well organised.

To sum up, our lightweight-yet-powerful architecture consists of RDCL and DCCL. The former is designed to achieve real-time speed on the CPU devices. The latter aims at enriching the receptive fields and combining coarse-to-fine information across different layers to handle faces of various scales.

Receptive Field	$75 \times 75,\ 107 \times 107,\ 139 \times 139,\ 171 \times 171,\ 203 \times 203,\ 235 \times 235$
Default Anchor	$16 \times 16,\ 32 \times 32,\ 64 \times 64,\ 128 \times 128,\ 256 \times 256$

Table 1. The receptive field of the last convolutional layer and the default anchor of our DCFPN.

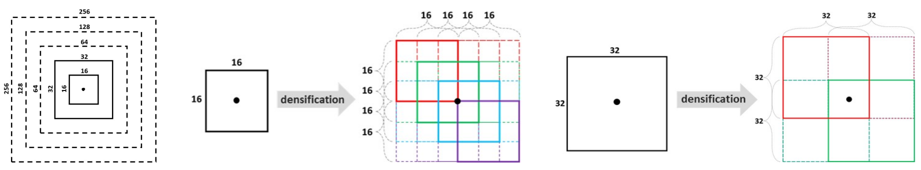


Fig. 2. Some illustrations of anchors. (a) 5 default anchors at a center of receptive filed. (b) 16×16 anchor densification. (c) 32×32 anchor densification. Best viewed in color.

3.2 Dense anchor strategy

As listed in Tab. 1, we use 5 default anchors that are associated with the last convolutional layer. Hence, these 5 default anchors have the same tiling interval on the image (*i.e.,* 16 pixels). It is obviously that there is a tiling density imbalance problem. Comparing with large anchors (i.e., 64×64, 128×128 and 256×256), small anchors (i.e., 16×16 and 32×32) are too sparse, which results in low recall rate of small faces.

To improve the recall rate of small faces, we use the dense anchor strategy proposed by [6] for small anchor. Specifically, without this dense anchor strategy, there are 5 anchors for every center of the receptive filed (Fig. 2(a)). To densify one type of anchors, this strategy uniformly tiles several anchors around the center of one receptive field instead of only tiling one. As illustrated in Fig. 2(b) and Fig. 2(c), the sampling interval of 16×16 and 32×32 anchor are densified to 4 and 8 pixels, respectively. Consequently, for every center of the receptive filed, there are total 23 anchors (16 from 16×16 anchor, 4 from 32×32 anchor and 3 from the rest three anchors). This dense anchor strategy is crucial to detect small faces.

3.3 Fair L1 loss

Our model is jointly optimized by two loss functions, L_{cls} and L_{reg}, which compute errors of score and coordinate, respectively. We adopt a 2-class softmax loss for L_{cls}. As for L_{reg}, to locate small faces well, we propose the fair L1 loss that directly regresses the predicted box's relative center coordinate and its width and height as follows:

$$\begin{array}{llll} t_x = x - x^a, & t_y = y - y^a, & t_w = w, & t_h = h \\ t_x^* = x^* - x^a, & t_y^* = y^* - y^a, & t_w^* = w^*, & t_h^* = h^* \end{array} \tag{1}$$

where x, y, w, and h denote the center coordinates and width and height. Variables x, x^a, and x^* are for the predicted box, anchor box, and GT box (likewise for y,w,h). The scale normalization is implemented to have scale-invariance loss value as follows:

$$L_{reg}(t, t^*) = \sum_{j \in \{x,y,w,h\}} fair_{L_1}(t_j - t_j^*) \tag{2}$$

in which

$$fair_{L_1}(z_j) = \begin{cases} |z_j| \;/\; gt_w & if\, j \in \{x,w\} \\ |z_j| \;/\; gt_h & otherwise \end{cases} \tag{3}$$

where gt_w and gt_h denote the GT box's width and height. It equally treats small and big face by directly regressing box's relative center coordinate and its width and height.

3.4 Training and implementation details

The DCFPN is trained end-to-end by the stochastic gradient descent (SGD) as follows:

Training labels. Face detection is a face and non-face classification task, and a binary label (*i.e.,* the positive or negative label) need to be assigned to each anchor during the training stage. The positive anchor is defined by the following two conditions: (i) Matching each face to the anchor with the best jaccard overlap; (ii) Matching anchors to any face with jaccard overlap higher than a threshold (usually 0.5). Anchors that do not be matched by the two conditions are negative anchors.

Training data. Our model is trained on 12880 images from the WIDER FACE [27] training set. To enrich the training dataset, each training image is sequentially processed by the color distortion, random cropping, scale transformation and horizontal flipping, eventually getting a 512×512 square sub-image from original image. The groundtruth bounding box is ignored if its center coordinate is located outside of the square sub-image. In the training process, each mini-batch is collected randomly from 48 images. For each mini-batch, all of the positive anchors and half of the negative anchors are used to train our model.

Implementation details. We randomly initialize all layers by drawing weights from a zero-mean Gaussian distribution with standard deviation 0.01. We use 0.9 momentum and 0.0005 weight decay. The maximum number of iterations is $100k$, and the initial learning rate is set to 0.1 and multiplied by 0.1 every $20k$ iterations. Our model is implemented in Caffe framework [28].

4 Experiments

In this section, we firstly analyze our model in an ablative way, then evaluate it on the common face detection benchmarks, finally introduce its runtime efficiency.

4.1 Model analysis

We carry out extensive ablation experiments on the FDDB dataset to analyze our model. For all the experiments, we use the same settings, except for specified changes to the components. To better understand DCFPN, we ablate each component one after another to examine how each component affects the final performance. Firstly, we replace the fair L1 loss with smooth L1 loss. Meantime, the target of regression is the same as RPN. Secondly, we ablate the dense anchor strategy [6]. Finally, we take the place of DCCL with four convolutional layers, which all have 3×3 kernel size and whose output number are 64, 128, 192 and 256, respectively.

Some promising conclusions can be summed up according to the ablative results listed in Tab. 2. Firstly, the comparison between the first and second columns in Tab. 2 indicates that the fair L1 loss function effectively increases the mAP performance by 0.7%, owning to locating small faces well. Secondly, ablating the dense anchor strategy results in 0.8% decline, showing the importance of this strategy. Finally, the DCCL is used to enrich the receptive fields and combine coarse-to-fine information across different layers to handle faces of various scales. From the results listed in Tab. 2, we can observe that the mAP on FDDB is reduced from 93.7% to 93.2% after replacing the DCCL. The sharp decline (*i.e.*, 0.5%) demonstrates the effectiveness of the DCCL.

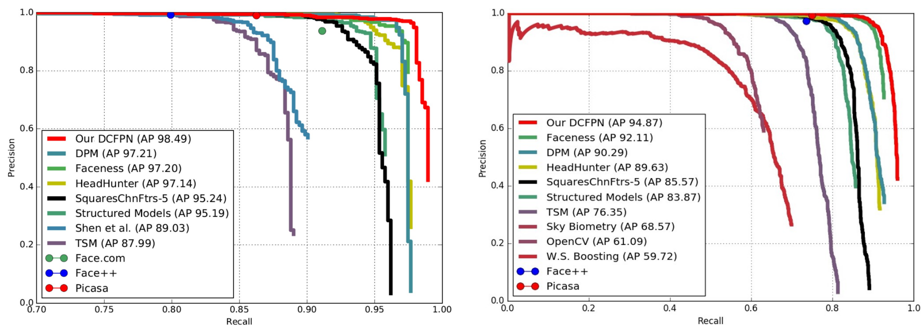
Component	DCFPN			
Designed architecture?	✓	✓	✓	
Dense anchor strategy [6]?	✓	✓		
Fair L1 loss?	✓			
Accuracy (mAP)	95.2	94.5	93.7	93.2

Table 2. Ablative results on FDDB. Accuracy means the true positive rate at 1000 false positives.

4.2 Evaluation on benchmark

This section presents the face detection bechmarking using our proposed DCFPN approach. We compare our results with those of other leading methods.

AFW database [22]. It contains 205 images with 473 labeled faces from Flickr. We evaluate our detector on this dataset and compare with well known research and commercial face detectors. Research detectors include [19,21,22,24,29]. Commercial detectors include Face.com, Face++ and Google Picasa. As can be observed from Fig. 3(a), our method outperforms strong all others by a large margin.

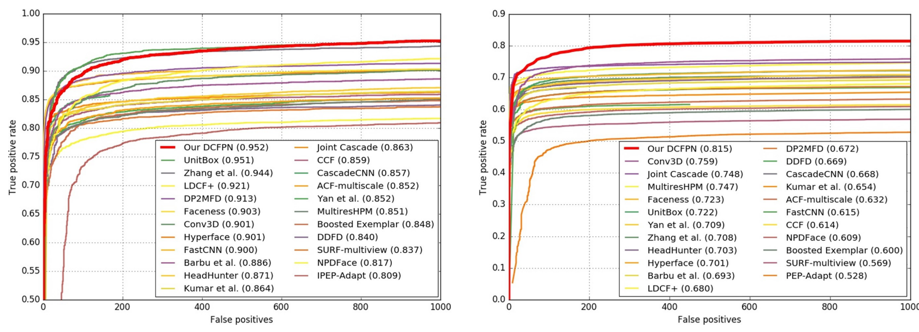


(a) AFW dataset (b) PASCAL faces dataset

Fig. 3. Precision-recall curves.

PASCAL face database [21]. It consists of 851 images with 1335 labeled faces and is collected from the test set of PASCAL person layout dataset, which is a subset of PASCAL VOC. There are large face appearance and pose variations in this dataset. Note that this dataset is designed for person layout detection and head annotation is used as face annotation. The cases when the face is occluded are common. Fig. 3(b) shows that our DCFPN method outperforms all other detectors.

FDDB database [30]. It has 5,171 faces in 2,845 images taken from news articles on Yahoo websites. FDDB uses ellipse face annotations while our DCFPN outputs rectangle outputs. This inconsistency has a great impact to the continuous score. For a more fair comparison under the continuous score evaluation, we regress a transformation matrix according to the ellipse and rectangle annotations, and then transform our rectangle outputs to ellipse outputs. As shown in Fig. 4(a) and Fig. 4(b), our DCFPN performs better than all of the published face detection methods, demonstrating that DCFPN is able to robustly detect unconstrained faces.



(a) Discontinuous score curves (b) Continuous score curves

Fig. 4. Evaluation on the FDDB dataset.

4.3 Runtime efficiency

CNN based methods have always been accused of their runtime efficiency. Recent CNN algorithms are getting faster on high-end GPUs. However, in most practical applications, especially CPU based applications, they are not fast enough.

Our DCFPN is efficient and accurate enough to meet practical requirements. Specifically, due to the great ability to detect small faces, the proposed DCFPN can shrink the test images by a few times and detect small faces, so as to reach real-time speed as well as maintain high performance. This means that faces can be efficiently detected by shrinking the test image and detecting smaller ones. With this advantage, our method can detect faces bigger than 40 pixels at 30 FPS on a 2.60GHz CPU for the VGA-resolution images. Besides, our method with only 3.2M parameter can directly run on a GPU card at 250 FPS for the VGA-resolution images.

5 Conclusion

In this paper, we propose a novel face detector with real-time speed on the CPU devices as well as high performance. On the one hand, our DCFPN has a lightweight-but-powerful framework that can well incorporate CNN features from different sizes of receptive field at multiple levels of abstraction. On the other hand, we use the dense anchor strategy and propose the fair L1 loss function to handle small faces well. The state-of-the-art performance on three challenge datasets shows its ability to detect faces in the uncontrolled environment. The proposed detector is very fast, achieving 30 FPS to detect faces bigger than 40 pixels on CPU and can be accelerated to 250 FPS on GPU for the VGA-resolution images.

Acknowledgments

This work was supported by the National Key Research and Development Plan (Grant No.2016YFC0801002).

References

1. Viola, P., Jones, M.J.: Robust real-time face detection. IJCV (2004)
2. Zhang, C., Zhang, Z.: A survey of recent advances in face detection. Technical Report (2010)
3. Lecun, Y., Bengio, Y.: Convolutional networks for images, speech, and time-series. The Handbook of Brain Theory and Neural Networks (1995)
4. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. CVPR (2014)
5. Li, H., Lin, Z., Shen, X., Brandt, J., Hua, G.: A convolutional neural network cascade for face detection. CVPR (2015)
6. Zhang, S., Zhu, X., Lei, Z., Shi, H., Wang, X., Li, S.Z.: Faceboxes: A cpu real-time face detector with high accuracy. IJCB (2017)
7. Yang, M.H., Kriegman, D.J., Ahuja, N.: Detecting faces in images: A survey. PAMI (2002)
8. Zafeiriou, S., Zhang, C., Zhang, Z.: A survey on face detection in the wild: Past, present and future. Computer Vision and Image Understanding (2015)
9. Yang, B., Yan, J., Lei, Z., Li, S.Z.: Aggregate channel features for multi-view face detection. IJCB (2014)
10. Zhang, L., Chu, R., Xiang, S., Liao, S., Li, S.Z.: Face detection based on multi-block lbp representation. ICB (2007)
11. Huang, C., Ai, H., Li, Y., Lao, S.: High-performance rotation invariant multiview face detection. PAMI (2007)
12. Jones, M., Viola, P.: Fast multi-view face detection. MERL (2003)
13. Zhang, C., Platt, J.C., Viola, P.A.: Multiple instance boosting for object detection. NIPS (2005)
14. Bourdev, L., Brandt, J.: Robust object detection via soft cascade. CVPR (2005)
15. Li, S.Z., Zhu, L., Zhang, Z., Blake, A., Zhang, H., Shum, H.: Statistical learning of multi-view face detection. ECCV (2002)
16. Xiao, R., Zhu, L., Zhang, H.J.: Boosting chain learning for object detection. ICCV (2003)
17. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. PAMI (2010)
18. Ghiasi, G., Fowlkes, C.C.: Occlusion coherence: Detecting and localizing occluded faces. arXiv preprint arXiv:1506.08347 (2015)
19. Mathias, M., Benenson, R., Pedersoli, M., Van Gool, L.: Face detection without bells and whistles. ECCV (2014)
20. Yan, J., Lei, Z., Wen, L., Li, S.Z.: The fastest deformable part model for object detection. CVPR (2014)
21. Yan, J., Zhang, X., Lei, Z., Li, S.Z.: Face detection by structural models. Image and Vision Computing (2014)
22. Zhu, X., Ramanan, D.: Face detection, pose estimation, and landmark localization in the wild. CVPR (2012)
23. Yang, B., Yan, J., Lei, Z., Li, S.Z.: Convolutional channel features. ICCV (2015)
24. Yang, S., Luo, P., Loy, C.C., Tang, X.: From facial parts responses to face detection: A deep learning approach. ICCV (2015)
25. Chen, D., Hua, G., Wen, F., Sun, J.: Supervised transformer network for efficient face detection. ECCV (2016)
26. Zhang, K., Zhang, Z., Li, Z., Qiao, Y.: Joint face detection and alignment using multi-task cascaded convolutional networks. arXiv preprint arXiv:1604.02878 (2016)
27. Yang, S., Luo, P., Loy, C., Tang, X.: Wider face: A face detection benchmark. CVPR (2016)
28. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. ACMMM (2014)
29. Shen, X., Lin, Z., Brandt, J., Wu, Y.: Detecting and aligning faces by image retrieval. CVPR (2013)
30. Jain, V., Learned-Miller, E.G.: Fddb: A benchmark for face detection in unconstrained settings. UMass Amherst Report (2010)