RetinaFace: Single-stage Dense Face Localisation in the Wild

Jiankang Deng * [1,2,4] Jia Guo * [2] Yuxiang Zhou [1]
Jinke Yu [2] Irene Kotsia [3] Stefanos Zafeiriou[1,4]
[1]Imperial College London [2]InsightFace [3]Middlesex University London [4]FaceSoft

Abstract

Though tremendous strides have been made in uncontrolled face detection, accurate and efficient face localisation in the wild remains an open challenge. This paper presents a robust single-stage face detector, named RetinaFace, which performs pixel-wise face localisation on various scales of faces by taking advantages of joint extra-supervised and self-supervised multi-task learning. Specifically, We make contributions in the following five aspects: (1) We manually annotate five facial landmarks on the WIDER FACE dataset and observe significant improvement in hard face detection with the assistance of this extra supervision signal. (2) We further add a self-supervised mesh decoder branch for predicting a pixel-wise 3D shape face information in parallel with the existing supervised branches. (3) On the WIDER FACE hard test set, RetinaFace outperforms the state of the art average precision (AP) by 1.1% *(achieving AP equal to* 91.4%*). (4) On the IJB-C test set, RetinaFace enables state of the art methods (ArcFace) to improve their results in face verification (TAR=*89.59% *for FAR=1e-6). (5) By employing light-weight backbone networks, RetinaFace can run real-time on a single CPU core for a VGA-resolution image. Extra annotations and code will be released to facilitate future research.*

1. Introduction

Automatic face localisation is the prerequisite step of facial image analysis for many applications such as facial attribute (*e.g.* expression [64] and age [38]) and facial identity recognition [45, 31, 55, 11]. A narrow definition of face localisation may refer to traditional face detection [53, 62], which aims at estimating the face bounding boxes without any scale and position prior. Nevertheless, in this paper we refer to a broader definition of face localisation which includes face detection [39], face alignment [13], pixel-wise face parsing [48] and 3D dense correspondence regression [2, 12]. That kind of dense face localisation provides accurate facial position information for all different scales.

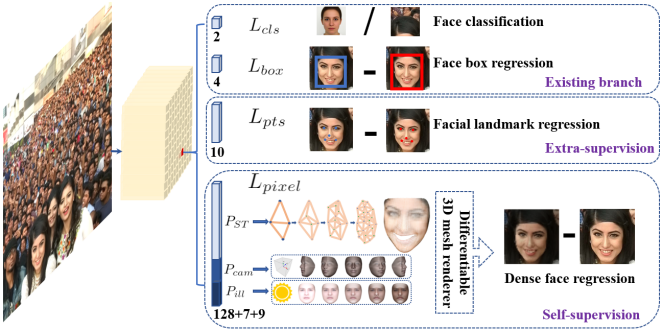


Figure 1. The proposed single-stage pixel-wise face localisation method employs extra-supervised and self-supervised multi-task learning in parallel with the existing box classification and regression branches. Each positive anchor outputs (1) a face score, (2) a face box, (3) five facial landmarks, and (4) dense 3D face vertices projected on the image plane.

Inspired by generic object detection methods [16, 43, 30, 41, 42, 28, 29], which embraced all the recent advances in deep learning, face detection has recently achieved remarkable progress [23, 36, 68, 8, 49]. Different from generic object detection, face detection features smaller ratio variations (from 1:1 to 1:1.5) but much larger scale variations (from several pixels to thousand pixels). The most recent state-of-the-art methods [36, 68, 49] focus on single-stage [30, 29] design which densely samples face locations and scales on feature pyramids [28], demonstrating promising performance and yielding faster speed compared to two-stage methods [43, 63, 8]. Following this route, we improve the single-stage face detection framework and propose a state-of-the-art dense face localisation method by exploiting multi-task losses coming from strongly supervised and self-supervised signals. Our idea is examplified in Fig. 1.

Typically, face detection training process contains both classification and box regression losses [16]. Chen *et al.* [6] proposed to combine face detection and alignment in a joint cascade framework based on the observation that aligned face shapes provide better features for face classification. Inspired by [6], MTCNN [66] and STN [5] simultaneously

* Equal contributions.
Email: j.deng16@imperial.ac.uk; guojia@gmail.com
InsightFace is a nonprofit Github project for 2D and 3D face analysis.

detected faces and five facial landmarks. Due to training data limitation, JDA [6], MTCNN [66] and STN [5] have not verified whether tiny face detection can benefit from the extra supervision of five facial landmarks. One of the questions we aim at answering in this paper is whether we can push forward the current best performance (90.3% [67]) on the WIDER FACE hard test set [60] by using extra supervision signal built of five facial landmarks.

In Mask R-CNN [20], the detection performance is significantly improved by adding a branch for predicting an object mask in parallel with the existing branch for bounding box recognition and regression. That confirms that dense pixel-wise annotations are also beneficial to improve detection. Unfortunately, for the challenging faces of WIDER FACE it is not possible to conduct dense face annotation (either in the form of more landmarks or semantic segments). Since supervised signals cannot be easily obtained, the question is whether we can apply unsupervised methods to further improve face detection.

In FAN [56], an anchor-level attention map is proposed to improve the occluded face detection. Nevertheless, the proposed attention map is quite coarse and does not contain semantic information. Recently, self-supervised 3D morphable models [14, 51, 52, 70] have achieved promising 3D face modelling in-the-wild. Especially, Mesh Decoder [70] achieves over real-time speed (over 2500FPS on CPU) by exploiting graph convolutions [10, 40] on joint shape and texture. However, the main challenges of applying mesh decoder [70] into the single-stage detector are: (1) camera parameters are hard to estimate accurately, and (2) the joint latent shape and texture representation is predicted from a single feature vector (1×1 Conv on feature pyramid) instead of the RoI pooled feature, which indicates the risk of feature shift. In this paper, we employ a mesh decoder [70] branch through self-supervision learning for predicting a pixel-wise 3D face shape in parallel with the existing supervised branches.

To summarise, our key contributions are:

- Based on a single-stage design, we propose a novel pixel-wise face localisation method named RetinaFace, which employs a multi-task learning strategy to simultaneously predict face score, face box, five facial landmarks, and 3D position and correspondence of each facial pixel.
- On the WIDER FACE hard subset, RetinaFace outperforms the AP of the state of the art two-stage method (ISRN [67]) by 1.1% (AP equal to 91.4%).
- On the IJB-C dataset, RetinaFace helps to improve ArcFace's [11] verification accuracy (with TAR equal to 89.59% when FAR=1e-6). This indicates that better face localisation can significantly improve face recognition.
- By employing light-weight backbone networks, RetinaFace can run real-time on a single CPU core for a VGA-resolution image.
- Extra annotations and code will be released to public.

2. Related Work

Image pyramid v.s. feature pyramid: The sliding-window paradigm, in which a classifier is applied on a dense image grid, can be traced back to past decades. The milestone work of Viola-Jones [53] explored cascade chain to reject false face regions from an image pyramid with real-time efficiency, leading to the widespread adoption of such scale-invariant face detection framework [66, 5]. Even though the sliding-window on image pyramid was the leading detection paradigm [19, 32], with the emergence of feature pyramid [28], sliding-anchor [43] on multi-scale feature maps [68, 49], quickly dominated face detection.

Two-stage v.s. single-stage: Current face detection methods have inherited some achievements from generic object detection approaches and can be divided into two categories: two-stage methods (*e.g.* Faster R-CNN [43, 63, 72]) and single-stage methods (*e.g.* SSD [30, 68] and RetinaNet [29, 49]). Two-stage methods employed a "proposal and refinement" mechanism featuring high localisation accuracy. By contrast, single-stage methods densely sampled face locations and scales, which resulted in extremely unbalanced positive and negative samples during training. To handle this imbalance, sampling [47] and re-weighting [29] methods were widely adopted. Compared to two-stage methods, single-stage methods are more efficient and have higher recall rate but at the risk of achieving a higher false positive rate and compromising the localisation accuracy.

Context Modelling: To enhance the model's contextual reasoning power for capturing tiny faces [23], SSH [36] and PyramidBox [49] applied context modules on feature pyramids to enlarge the receptive field from Euclidean grids. To enhance the non-rigid transformation modelling capacity of CNNs, deformable convolution network (DCN) [9, 74] employed a novel deformable layer to model geometric transformations. The champion solution of the WIDER Face Challenge 2018 [33] indicates that rigid (expansion) and non-rigid (deformation) context modelling are complementary and orthogonal to improve the performance of face detection.

Multi-task Learning: Joint face detection and alignment is widely used [6, 66, 5] as aligned face shapes provide better features for face classification. In Mask R-CNN [20], the detection performance was significantly improved by adding a branch for predicting an object mask in parallel with the existing branches. Densepose [1] adopted the architecture of Mask-RCNN to obtain dense part labels and coordinates within each of the selected regions. Nevertheless, the dense regression branch in [20, 1] was trained by supervised learning. In addition, the dense branch was a

small FCN applied to each RoI to predict a pixel-to-pixel dense mapping.

3. RetinaFace

3.1. Multi-task Loss

For any training anchor i, we minimise the following multi-task loss:

$$L = L_{cls}(p_i, p_i^*) + \lambda_1 p_i^* L_{box}(t_i, t_i^*) + \lambda_2 p_i^* L_{pts}(l_i, l_i^*) + \lambda_3 p_i^* L_{pixel}. \quad (1)$$

(1) Face classification loss $L_{cls}(p_i, p_i^*)$, where p_i is the predicted probability of anchor i being a face and p_i^* is 1 for the positive anchor and 0 for the negative anchor. The classification loss L_{cls} is the softmax loss for binary classes (face/not face). (2) Face box regression loss $L_{box}(t_i, t_i^*)$, where $t_i = \{t_x, t_y, t_w, t_h\}_i$ and $t_i^* = \{t_x^*, t_y^*, t_w^*, t_h^*\}_i$ represent the coordinates of the predicted box and ground-truth box associated with the positive anchor. We follow [16] to normalise the box regression targets (*i.e.* centre location, width and height) and use $L_{box}(t_i, t_i^*) = R(t_i - t_i^*)$, where R is the robust loss function (smooth-L_1) defined in [16]. (3) Facial landmark regression loss $L_{pts}(l_i, l_i^*)$, where $l_i = \{l_{x_1}, l_{y_1}, \ldots, l_{x_5}, l_{y_5}\}_i$ and $l_i^* = \{l_{x_1}^*, l_{y_1}^*, \ldots, l_{x_5}^*, l_{y_5}^*\}_i$ represent the predicted five facial landmarks and ground-truth associated with the positive anchor. Similar to the box centre regression, the five facial landmark regression also employs the target normalisation based on the anchor centre. (4) Dense regression loss L_{pixel} (refer to Eq. 3). The loss-balancing parameters λ_1-λ_3 are set to 0.25, 0.1 and 0.01, which means that we increase the significance of better box and landmark locations from supervision signals.

3.2. Dense Regression Branch

Mesh Decoder. We directly employ the mesh decoder (mesh convolution and mesh up-sampling) from [70, 40], which is a graph convolution method based on fast localised spectral filtering [10]. In order to achieve further acceleration, we also use a joint shape and texture decoder similarly to the method in [70], contrary to [40] which only decoded shape.

Below we will briefly explain the concept of graph convolutions and outline why they can be used for fast decoding. As illustrated in Fig. 3(a), a 2D convolutional operation is a "kernel-weighted neighbour sum" within the Euclidean grid receptive field. Similarly, graph convolution also employs the same concept as shown in Fig. 3(b). However, the neighbour distance is calculated on the graph by counting the minimum number of edges connecting two vertices. We follow [70] to define a coloured face mesh $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} \in \mathbb{R}^{n \times 6}$ is a set of $n = 28,431$ face vertices containing the joint shape and texture information, and $\mathcal{E} \in \{0,1\}^{n \times n}$ is a **sparse** adjacency matrix encoding the connection status between vertices. The graph Laplacian is defined as $L = D - \mathcal{E} \in \mathbb{R}^{n \times n}$ where $D \in \mathbb{R}^{n \times n}$ is a diagonal matrix with $D_{ii} = \sum_j \mathcal{E}_{ij}$.

Following [10, 40, 70], the graph convolution with kernel g_θ can be formulated as a recursive Chebyshev polynomial truncated at order K,

$$y = g_\theta(L)x = \sum_{k=0}^{K-1} \theta_k T_k(\tilde{L})x, \quad (2)$$

where $\theta \in \mathbb{R}^K$ is a vector of Chebyshev coefficients and $T_k(\tilde{L}) \in \mathbb{R}^{n \times n}$ is the Chebyshev polynomial of order k evaluated at the scaled Laplacian $\tilde{L}$. Denoting $\bar{x}_k = T_k(\tilde{L})x \in \mathbb{R}^n$, we can recurrently compute $\bar{x}_k = 2\tilde{L}\bar{x}_{k-1} - \bar{x}_{k-2}$ with $\bar{x}_0 = x$ and $\bar{x}_1 = \tilde{L}x$. The whole filtering operation is extremely efficient including K **sparse** matrix-vector multiplications and one dense matrix-vector multiplication $y = g_\theta(L)x = [\bar{x}_0, \ldots, \bar{x}_{K-1}]\theta$.

Differentiable Renderer. After we predict the shape and texture parameters $P_{ST} \in \mathbb{R}^{128}$, we employ an efficient differentiable 3D mesh renderer [14] to project a coloured-mesh $\mathcal{D}_{P_{ST}}$ onto a 2D image plane with camera parameters $P_{cam} = [x_c, y_c, z_c, x_c', y_c', z_c', f_c]$ (*i.e.* camera location, camera pose and focal length) and illumination parameters $P_{ill} = [x_l, y_l, z_l, r_l, g_l, b_l, r_a, g_a, b_a]$ (*i.e.* location of point light source, colour values and colour of ambient lighting).

Dense Regression Loss. Once we get the rendered 2D face $\mathcal{R}(\mathcal{D}_{P_{ST}}, P_{cam}, P_{ill})$, we compare the pixel-wise difference of the rendered and the original 2D face using the following function:

$$L_{pixel} = \frac{1}{W * H} \sum_i^W \sum_j^H \| \mathcal{R}(\mathcal{D}_{P_{ST}}, P_{cam}, P_{ill})_{i,j} - I_{i,j}^* \|_1, \quad (3)$$

where W and H are the width and height of the anchor crop $I_{i,j}^*$, respectively.

4. Experiments

4.1. Dataset

The WIDER FACE dataset [60] consists of $32,203$ images and $393,703$ face bounding boxes with a high degree of variability in scale, pose, expression, occlusion and illumination. The WIDER FACE dataset is split into training (40%), validation (10%) and testing (50%) subsets by randomly sampling from 61 scene categories. Based on the detection rate of EdgeBox [76], three levels of difficulty (*i.e.* Easy, Medium and Hard) are defined by incrementally incorporating hard samples.

Extra Annotations. As illustrated in Fig. 4 and Tab. 1, we define five levels of face image quality (according to how difficult it is to annotate landmarks on the face) and annotate five facial landmarks (*i.e.* eye centres, nose tip and mouth

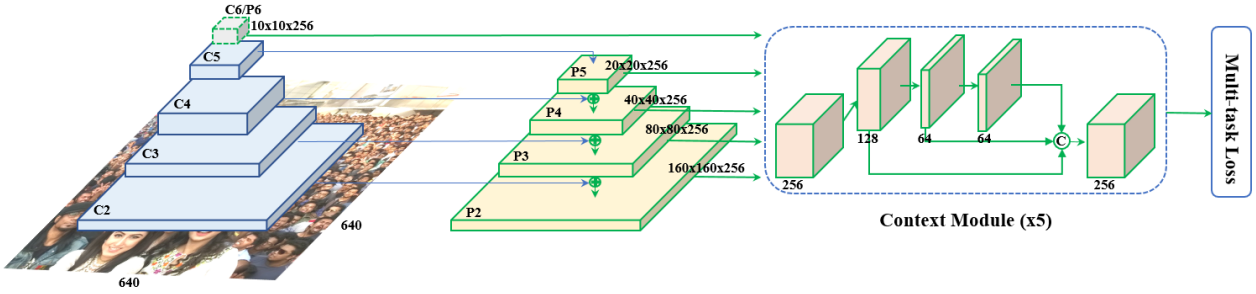
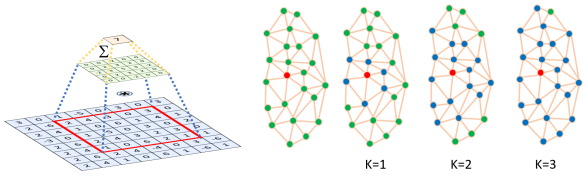


Figure 2. An overview of the proposed single-stage dense face localisation approach. RetinaFace is designed based on the feature pyramids with independent context modules. Following the context modules, we calculate a multi-task loss for each anchor.



(a) 2D Convolution (b) Graph Convolution

Figure 3. (a) 2D Convolution is kernel-weighted neighbour sum within the Euclidean grid receptive field. Each convolutional layer has $Kernel_H \times Kernel_W \times Channel_{in} \times Channel_{out}$ parameters. (b) Graph convolution is also in the form of kernel-weighted neighbour sum, but the neighbour distance is calculated on the graph by counting the minimum number of edges connecting two vertices. Each convolutional layer has $K \times Channel_{in} \times Channel_{out}$ parameters and the Chebyshev coefficients $\theta_{i,j} \in \mathbb{R}^K$ are truncated at order K.

corners) on faces that can be annotated from the WIDER FACE training and validation subsets. In total, we have annotated $84.6k$ faces on the training set and $18.5k$ faces on the validation set.

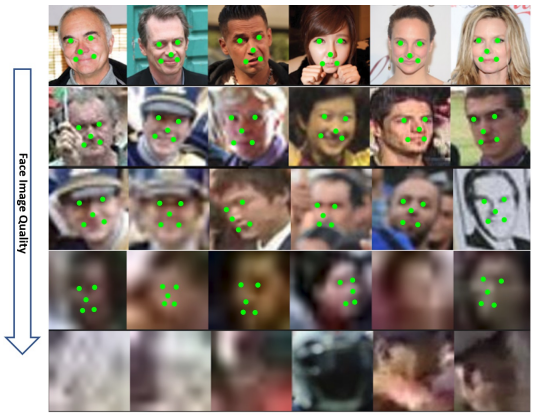


Figure 4. We add extra annotations of five facial landmarks on faces that can be annotated (we call them "annotatable") from the WIDER FACE training and validation sets.

4.2. Implementation details

Feature Pyramid. RetinaFace employs feature pyramid levels from P_2 to P_6, where P_2 to P_5 are computed from the output of the corresponding ResNet residual stage (C_2 through C_5) using top-down and lateral connections as in [28, 29]. P_6 is calculated through a 3×3 convolution with stride=2 on C_5. C_1 to C_5 is a pre-trained ResNet-152 [21] classification network on the ImageNet-11k dataset while P_6 are randomly initialised with the "Xavier" method [17].

Level	Face Number	Criterion
1	4,127	indisputable 68 landmarks [44]
2	12,636	annotatable 68 landmarks [44]
3	38,140	indisputable 5 landmarks
4	50,024	annotatable 5 landmarks
5	94,095	distinguish by context

Table 1. Five levels of face image quality. In the indisputable category a human can, without a lot of effort, locale the landmarks. In the annotatable category finding an approximate location requires some effort.

Context Module. Inspired by SSH [36] and PyramidBox [49], we also apply independent context modules on five feature pyramid levels to increase the receptive field and enhance the rigid context modelling power. Drawing lessons from the champion of the WIDER Face Challenge 2018 [33], we also replace all 3×3 convolution layers within the lateral connections and context modules by the deformable convolution network (DCN) [9, 74], which further strengthens the non-rigid context modelling capacity.

Loss Head. For negative anchors, only classification loss is applied. For positive anchors, the proposed multi-task loss is calculated. We employ a shared loss head (1×1 conv) across different feature maps $H_n \times W_n \times 256, n \in \{2, \ldots, 6\}$. For the mesh decoder, we directly apply the pre-trained model [70], which is a small computational overhead that allows for real-time running on a CPU or even on an ARM system.

Anchor Settings. As illustrated in Tab. 2, we employ scale-specific anchors on the feature pyramid levels from P_2 to P_6 like [56]. Here, P_2 is designed to capture tiny faces by tiling small anchors at the cost of more computational time and at the risk of more false positives. We set the scale step at $2^{1/3}$ and the aspect ratio at 1:1. With the input image size at 640×640, the anchors can cover scales from 16×16 to 406×406 on the feature pyramid levels. In total, there are 102,300 anchors, and 75% of these anchors are from P_2.

Feature Pyramid	Stride	Anchor
P_2 ($160 \times 160 \times 256$)	4	16, 20.16, 25.40
P_3 ($80 \times 80 \times 256$)	8	32, 40.32, 50.80
P_4 ($40 \times 40 \times 256$)	16	64, 80.63, 101.59
P_5 ($20 \times 20 \times 256$)	32	128, 161.26, 203.19
P_6 ($10 \times 10 \times 256$)	64	256, 322.54, 406.37

Table 2. The details of feature pyramid, stride size, anchor in RetinaFace. For a 640×640 input image, there are 102,300 anchors in total, and 75% of these anchors are tiled on P_2.

During training, anchors are matched to a ground-truth box when IoU is larger than 0.5, and to the background when IoU is less than 0.3. Unmatched anchors are ignored during training. Since most of the anchors ($> 99\%$) are negative after the matching step, we employ standard OHEM [47, 68] to alleviate significant imbalance between the positive and negative training examples. More specifically, we sort negative anchors by the loss values and select the top ones so that the ratio between the negative and positive samples is at least 3:1.

Data Augmentation. Since there are around 20% tiny faces in the WIDER FACE training set, we follow [68, 49] and randomly crop square patches from the original images and resize these patches into 640×640 to generate larger training faces. More specifically, square patches are cropped from the original image with a random size between [0.3, 1] of the smaller dimension of the original image. For the faces on the crop boundary, we keep the overlapped part of the face box if its centre is within the crop patch. Besides random crop, we also augment training data by random horizontal flip with the probability of 0.5 and photometric colour distortion [68].

Training Details. We train the RetinaFace using SGD optimiser (momentum at 0.9, weight decay at 0.0005, batch size of 8×4) on four NVIDIA Tesla P40 (24GB) GPUs. The learning rate starts from 10^{-3}, rising to 10^{-2} after 5 epochs, then divided by 10 at 55 and 68 epochs. The training process terminates at 80 epochs.

Testing Details. For testing on WIDER FACE, we follow the standard practices of [36, 68] and employ flip as well as multi-scale (the minimum image size at $[500, 800, 1100, 1400, 1700]$) strategies. Box voting [15] is applied on the union set of predicted face boxes using an IoU threshold at 0.4.

4.3. Ablation Study

To achieve a better understanding of the proposed RetinaFace, we conduct extensive ablation experiments to examine how the annotated five facial landmarks and the proposed dense regression branch quantitatively affect the performance of face detection. Besides the standard evaluation metric of average precision (AP) when IoU=0.5 on the Easy, Medium and Hard subsets, we also make use of the development server (Hard validation subset) of the WIDER Face Challenge 2018 [33], which employs a more strict evaluation metric of mean AP (mAP) for IoU=0.5:0.05:0.95, rewarding more accurate face detectors.

As illustrated in Tab. 3, we evaluate the performance of several different settings on the WIDER FACE validation set and focus on the observations of AP and mAP on the Hard subset. By applying the practices of state-of-the-art techniques (*i.e.* FPN, context module, and deformable convolution), we set up a strong baseline (91.286%), which is slightly better than ISRN [67] (90.9%). Adding the branch of five facial landmark regression significantly improves the face box AP (0.408%) and mAP (0.775%) on the Hard subset, suggesting that landmark localisation is crucial for improving the accuracy of face detection. By contrast, adding the dense regression branch increases the face box AP on Easy and Medium subsets but slightly deteriorates the results on the Hard subset, indicating the difficulty of dense regression under challenging scenarios. Nevertheless, learning landmark and dense regression jointly enables a further improvement compared to adding landmark regression only. This demonstrates that landmark regression does help dense regression, which in turn boosts face detection performance even further.

Method	Easy	Medium	Hard	mAP
FPN+Context	95.532	95.134	90.714	50.842
+DCN	96.349	95.833	91.286	51.522
+L_{pts}	96.467	96.075	91.694	52.297
+L_{pixel}	96.413	95.864	91.276	51.492
+$L_{pts} + L_{pixel}$	**96.942**	**96.175**	**91.857**	**52.318**

Table 3. Ablation experiments of the proposed methods on the WIDER FACE validation subset.

4.4. Face box Accuracy

Following the stander evaluation protocol of the WIDER FACE dataset, we only train the model on the training set and test on both the validation and test sets. To obtain the evaluation results on the test set, we submit the detection results to the organisers for evaluation. As shown in Fig. 5, we compare the proposed RetinaFace with other 24 state-of-the-art face detection algorithms (*i.e.* Multi-scale Cascade CNN [60], Two-stage CNN [60], ACF-WIDER [58], Faceness-WIDER [59], Multitask Cascade CNN [66], CMS-RCNN [72], LDCF+ [37], HR [23], Face R-CNN [54], ScaleFace [61], SSH [36], SFD [68], Face R-FCN [57], MSCNN [4], FAN [56], Zhu *et al*. [71], PyramidBox [49], FDNet [63], SRN [8], FANet [65], DSFD [27], DFS [50], VIM-FD [69], ISRN [67]). Our approach outperforms these state-of-the-art methods in terms of AP. More specifically, RetinaFace produces the best AP in all subsets of both validation and test sets, *i.e.*, 96.9% (Easy), 96.1% (Medium) and 91.8% (Hard) for validation set, and 96.3%

(Easy), 95.6% (Medium) and 91.4% (Hard) for test set. Compared to the recent best performed method [67], RetinaFace sets up a new impressive record (91.4% v.s. 90.3%) on the Hard subset which contains a large number of tiny faces.

In Fig. 6, we illustrate qualitative results on a selfie with dense faces. RetinaFace successfully finds about 900 faces (threshold at 0.5) out of the reported $1,151$ faces. Besides accurate bounding boxes, the five facial landmarks predicted by RetinaFace are also very robust under the variations of pose, occlusion and resolution. Even though there are some failure cases of dense face localisation under heavy occlusion, the dense regression results on some clear and large faces are good and even show expression variations.

4.5. Five Facial Landmark Accuracy

To evaluate the accuracy of five facial landmark localisation, we compare RetinaFace with MTCNN [66] on the AFLW dataset [26] (24,386 faces) as well as the WIDER FACE validation set (18.5k faces). Here, we employ the face box size ($\sqrt{W \times H}$) as the normalisation distance. As shown in Fig. 7(a), we give the mean error of each facial landmark on the AFLW dataset [73]. RetinaFace significantly decreases the normalised mean errors (NME) from 2.72% to 2.21% when compared to MTCNN. In Fig. 7(b), we show the cumulative error distribution (CED) curves on the WIDER FACE validation set. Compared to MTCNN, RetinaFace significantly decreases the failure rate from 26.31% to 9.37% (the NME threshold at 10%).

4.6. Dense Facial Landmark Accuracy

Besides box and five facial landmarks, RetinaFace also outputs dense face correspondence, but the dense regression branch is trained by self-supervision learning only. Following [12, 70], we evaluate the accuracy of dense facial landmark localisation on the AFLW2000-3D dataset [75] considering (1) 68 landmarks with the 2D projection coordinates and (2) all landmarks (28,431 in RetinaFace) with 3D coordinates. Here, the mean error is still normalised by the bounding box size [75]. In Fig. 8(a) and 8(b), we give the CED curves of state-of-the-art methods [12, 70, 75, 25, 3] as well as RetinaFace. Even though the performance gap exists between supervised and self-supervised methods, the dense regression results of RetinaFace are comparable with these state-of-the-art methods. More specifically, we observe that (1) five facial landmarks regression can alleviate the training difficulty of dense regression branch and significantly improve the dense regression results. (2) using single-stage features (as in RetinaFace) to predict dense correspondence parameters is much harder than employing (Region of Interest) RoI features (as in Mesh Decoder [70]). As illustrated in Fig. 8(c), RetinaFace can easily handle faces with pose variations but has difficulty under complex scenarios. This indicates that mis-aligned and over-compacted feature representation ($1 \times 1 \times 256$ in RetinaFace) impedes the single-stage framework achieving high accurate dense regression outputs. Nevertheless, the projected face regions in the dense regression branch still have the effect of attention [56] which can help to improve face detection as confirmed in the section of ablation study.

4.7. Face Recognition Accuracy

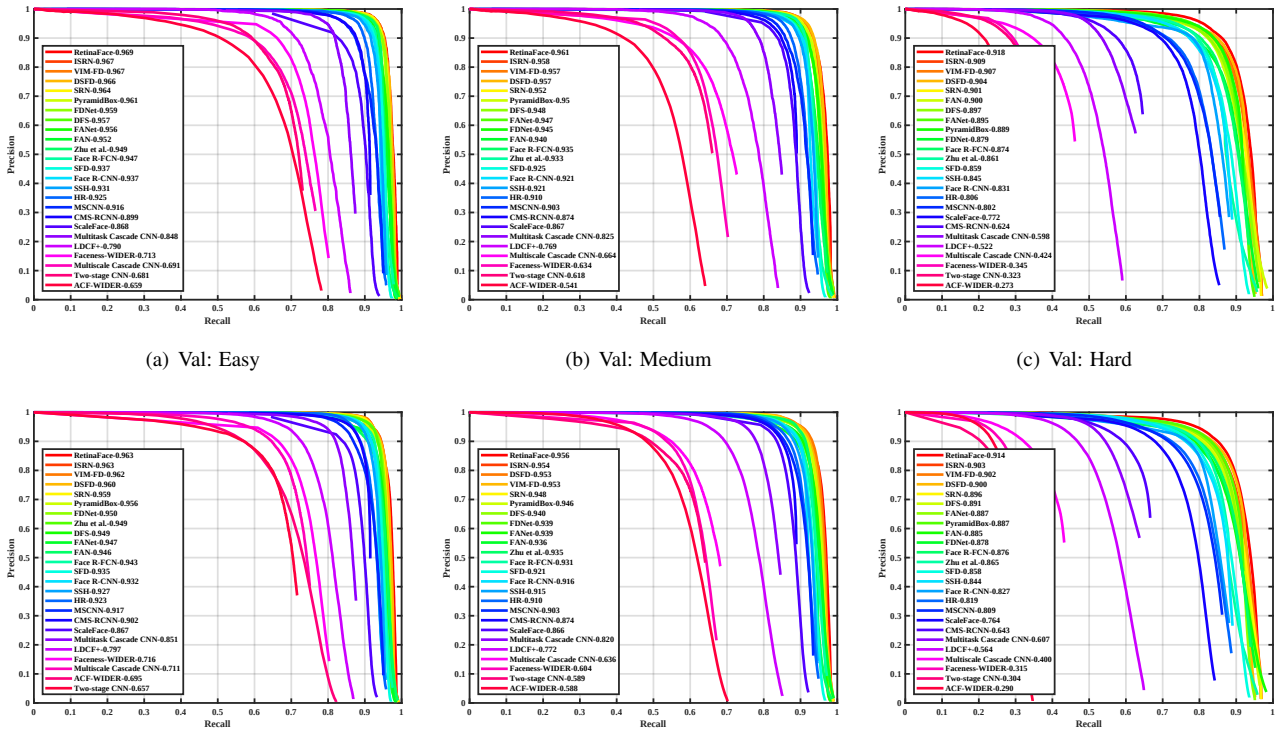
Face detection plays a crucial role in robust face recognition but its effect is rarely explicitly measured. In this paper, we demonstrate how our face detection method can boost the performance of a state-of-the-art publicly available face recognition method, *i.e*. ArcFace [11]. ArcFace [11] studied how different aspects in the training process of a deep convolutional neural network (*i.e*., choice of the training set, the network and the loss function) affect large scale face recognition performance. However, ArcFace paper did not study the effect of face detection by applying only the MTCNN [66] for detection and alignment. In this paper, we replace MTCNN by RetinaFace to detect and align all of the training data (*i.e*. MS1M [18]) and test data (*i.e*. LFW [24], CFP-FP [46], AgeDB-30 [35] and IJBC [34]), and keep the embedding network (*i.e*. ResNet100 [21]) and the loss function (*i.e*. additive angular margin) exactly the same as ArcFace.

In Tab. 4, we show the influence of face detection and alignment on deep face recognition (*i.e*. ArcFace) by comparing the widely used MTCNN [66] and the proposed RetinaFace. The results on CFP-FP, demonstrate that RetinaFace can boost ArcFace's verification accuracy from 98.37% to 99.49%. This result shows that the performance of frontal-profile face verification is now approaching that of frontal-frontal face verification (*e.g*. 99.86% on LFW).

Methods	LFW	CFP-FP	AgeDB-30
MTCNN+ArcFace [11]	99.83	98.37	98.15
RetinaFace+ArcFace	**99.86**	**99.49**	**98.60**

Table 4. Verification performance (%) of different methods on LFW, CFP-FP and AgeDB-30.

In Fig. 9, we show the ROC curves on the IJB-C dataset as well as the TAR for FAR$= 1e-6$ at the end of each legend. We employ two tricks (*i.e*. flip test and face detection score to weigh samples within templates) to progressively improve the face verification accuracy. Under fair comparison, TAR (at FAR$= 1e-6$) significantly improves from 88.29% to 89.59% simply by replacing MTCNN with RetinaFace. This indicates that (1) face detection and alignment significantly affect face recognition performance and (2) RetinaFace is a much stronger baseline that MTCNN for face recognition applications.

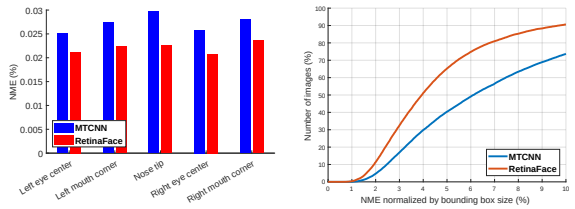


(a) Val: Easy (b) Val: Medium (c) Val: Hard

(d) Test: Easy (e) Test: Medium (f) Test: Hard

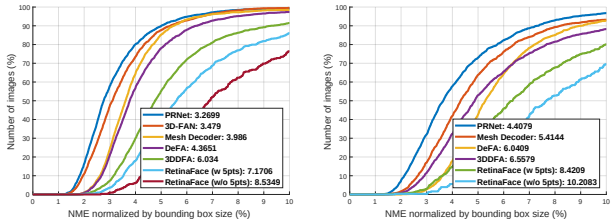
Figure 5. Precision-recall curves on the WIDER FACE validation and test subsets.

Figure 6. RetinaFace can find around 900 faces (threshold at 0.5) out of the reported 1151 people, by taking advantages of the proposed joint extra-supervised and self-supervised multi-task learning. Detector confidence is given by the colour bar on the right. Dense localisation masks are drawn in blue. Please zoom in to check the detailed detection, alignment and dense regression results on tiny faces.

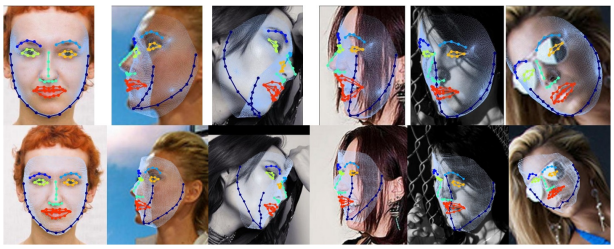


(a) NME on AFLW (b) CED on WIDER FACE

Figure 7. Qualitative comparison between MTCNN and RetinaFace on five facial landmark localisation. (a) AFLW (b) WIDER FACE validation set.



(a) 68 2D Landmarks (b) All 3D Landmarks



(c) Result Analysis (Upper: Mesh Decoder; Lower: RetinaFace)

Figure 8. CED curves on AFLW2000-3D. Evaluation is performed on (a) 68 landmarks with the 2D coordinates and (b) all landmarks with 3D coordinates. In (c), we compare the dense regression results from RetinaFace and Mesh Decoder [70]. RetinaFace can easily handle faces with pose variations but has difficulty to predict accurate dense correspondence under complex scenarios.

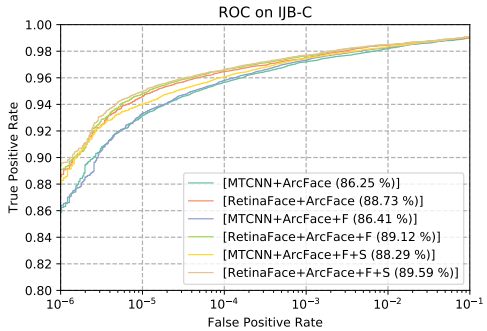


Figure 9. ROC curves of 1:1 verification protocol on the IJB-C dataset. "+F" refers to flip test during feature embedding and "+S" denotes face detection score used to weigh samples within templates. We also give TAR for FAR= $1e-6$ at the end of the each legend.

4.8. Inference Efficiency

During testing, RetinaFace performs face localisation in a single stage, which is flexible and efficient. Besides the above-explored heavy-weight model (ResNet-152, size of 262MB, and AP 91.8% on the WIDER FACE hard set), we also resort to a light-weight model (MobileNet-0.25 [22], size of 1MB, and AP 78.2% on the WIDER FACE hard set) to accelerate the inference.

For the light-weight model, we can quickly reduce the data size by using a 7×7 convolution with stride=4 on the input image, tile dense anchors on P_3, P_4 and P_5 as in [36], and remove deformable layers. In addition, the first two convolutional layers initialised by the ImageNet pre-trained model are fixed to achieve higher accuracy.

Tab. 5 gives the inference time of two models with respect to different input sizes. We omit the time cost on the dense regression branch, thus the time statistics are irrelevant to the face density of the input image. We take advantage of TVM [7] to accelerate the model inference and timing is performed on the NVIDIA Tesla P40 GPU, Intel i7-6700K CPU and ARM-RK3399, respectively. RetinaFace-ResNet-152 is designed for highly accurate face localisation, running at 13 FPS for VGA images (640×480). By contrast, RetinaFace-MobileNet-0.25 is designed for highly efficient face localisation which demonstrates considerable real-time speed of 40 FPS at GPU for 4K images (4096×2160), 20 FPS at multi-thread CPU for HD images (1920×1080), and 60 FPS at single-thread CPU for VGA images (640×480). Even more impressively, 16 FPS at ARM for VGA images (640×480) allows for a fast system on mobile devices.

Backbones	VGA	HD	4K
ResNet-152 (GPU)	75.1	443.2	1742
MobileNet-0.25 (GPU)	1.4	6.1	25.6
MobileNet-0.25 (CPU-m)	5.5	50.3	-
MobileNet-0.25 (CPU-1)	17.2	130.4	-
MobileNet-0.25 (ARM)	61.2	434.3	-

Table 5. Inference time (ms) of RetinaFace with different backbones (ResNet-152 and MobileNet-0.25) on different input sizes (VGA@640x480, HD@1920x1080 and 4K@4096x2160). "CPU-1" and "CPU-m" denote single-thread and multi-thread test on the Intel i7-6700K CPU, respectively. "GPU" refers to the NVIDIA Tesla P40 GPU and "ARM" platform is RK3399(A72x2).

5. Conclusions

We studied the challenging problem of simultaneous dense localisation and alignment of faces of arbitrary scales in images and we proposed the first, to the best of our knowledge, one-stage solution (RetinaFace). Our solution outperforms state of the art methods in the current most challenging benchmarks for face detection. Furthermore,

when RetinaFace is combined with state-of-the-art practices for face recognition it obviously improves the accuracy. The data and models will be provided publicly available to facilitate further research on the topic.

6. Acknowledgements

Jiankang Deng acknowledges financial support from the Imperial President's PhD Scholarship and GPU donations from NVIDIA. Stefanos Zafeiriou acknowledges support from EPSRC Fellowship DEFORM (EP/S010203/1), FACER2VM (EP/N007743/1) and a Google Faculty Fellowship.

References

[1] R. Alp Güler, N. Neverova, and I. Kokkinos. Densepose: Dense human pose estimation in the wild. In *CVPR*, 2018. 2

[2] R. Alp Guler, G. Trigeorgis, E. Antonakos, P. Snape, S. Zafeiriou, and I. Kokkinos. Densereg: Fully convolutional dense shape regression in-the-wild. In *CVPR*, 2017. 1

[3] A. Bulat and G. Tzimiropoulos. How far are we from solving the 2d & 3d face alignment problem?(and a dataset of 230,000 3d facial landmarks). In *ICCV*, 2017. 6

[4] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos. A unified multi-scale deep convolutional neural network for fast object detection. In *ECCV*, 2016. 5

[5] D. Chen, G. Hua, F. Wen, and J. Sun. Supervised transformer network for efficient face detection. In *ECCV*, 2016. 1, 2

[6] D. Chen, S. Ren, Y. Wei, X. Cao, and J. Sun. Joint cascade face detection and alignment. In *ECCV*, 2014. 1, 2

[7] T. Chen, T. Moreau, Z. Jiang, L. Zheng, E. Yan, H. Shen, M. Cowan, L. Wang, Y. Hu, L. Ceze, et al. Tvm: An automated end-to-end optimizing compiler for deep learning. In *OSDI*, 2018. 8

[8] C. Chi, S. Zhang, J. Xing, Z. Lei, S. Z. Li, and X. Zou. Selective refinement network for high performance face detection. *AAAI*, 2019. 1, 5

[9] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. In *ICCV*, 2017. 2, 4

[10] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NeurIPS*, 2016. 2, 3

[11] J. Deng, J. Guo, N. Xue, and S. Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *CVPR*, 2019. 1, 2, 6

[12] Y. Feng, F. Wu, X. Shao, Y. Wang, and X. Zhou. Joint 3d face reconstruction and dense alignment with position map regression network. In *ECCV*, 2018. 1, 6

[13] Z.-H. Feng, J. Kittler, M. Awais, P. Huber, and X.-J. Wu. Wing loss for robust facial landmark localisation with convolutional neural networks. In *CVPR*, 2018. 1

[14] K. Genova, F. Cole, A. Maschinot, A. Sarna, D. Vlasic, and W. T. Freeman. Unsupervised training for 3d morphable model regression. In *CVPR*, 2018. 2, 3

[15] S. Gidaris and N. Komodakis. Object detection via a multi-region and semantic segmentation-aware cnn model. In *ICCV*, 2015. 5

[16] R. Girshick. Fast r-cnn. In *ICCV*, 2015. 1, 3

[17] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, 2010. 4

[18] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *ECCV*, 2016. 6

[19] Z. Hao, Y. Liu, H. Qin, J. Yan, X. Li, and X. Hu. Scale-aware face detection. In *CVPR*, 2017. 2

[20] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *ICCV*, 2017. 2

[21] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4, 6

[22] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv:1704.04861*, 2017. 8

[23] P. Hu and D. Ramanan. Finding tiny faces. In *CVPR*, 2017. 1, 2, 5

[24] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, 2007. 6

[25] A. Jourabloo and X. Liu. Large-pose face alignment via cnn-based dense 3d model fitting. In *CVPR*, 2016. 6

[26] M. Koestinger, P. Wohlhart, P. M. Roth, and H. Bischof. Annotated facial landmarks in the wild: A large-scale, real-world database for facial landmark localization. In *ICCV workshops*, 2011. 6

[27] J. Li, Y. Wang, C. Wang, Y. Tai, J. Qian, J. Yang, C. Wang, J. Li, and F. Huang. Dsfd: dual shot face detector. *arXiv:1810.10220*, 2018. 5

[28] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 1, 2, 4

[29] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 1, 2, 4

[30] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016. 1, 2

[31] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song. Sphereface: Deep hypersphere embedding for face recognition. In *CVPR*, 2017. 1

[32] Y. Liu, H. Li, J. Yan, F. Wei, X. Wang, and X. Tang. Recurrent scale approximation for object detection in cnn. In *CVPR*, 2017. 2

[33] C. C. Loy, D. Lin, W. Ouyang, Y. Xiong, S. Yang, Q. Huang, D. Zhou, W. Xia, Q. Li, P. Luo, et al. Wider face and pedestrian challenge 2018: Methods and results. *arXiv:1902.06854*, 2019. 2, 4, 5

[34] B. Maze, J. Adams, J. A. Duncan, N. Kalka, T. Miller, C. Otto, A. K. Jain, W. T. Niggel, J. Anderson, J. Cheney, et al. Iarpa janus benchmark-c: Face dataset and protocol. In *ICB*, 2018. 6

[35] S. Moschoglou, A. Papaioannou, C. Sagonas, J. Deng, I. Kotsia, and S. Zafeiriou. Agedb: The first manually collected in-the-wild age database. In *CVPR Workshop*, 2017. 6

[36] M. Najibi, P. Samangouei, R. Chellappa, and L. S. Davis. Ssh: Single stage headless face detector. In *ICCV*, 2017. 1, 2, 4, 5, 8

[37] E. Ohn-Bar and M. M. Trivedi. To boost or not to boost? on the limits of boosted trees for object detection. In *ICPR*, 2016. 5

[38] H. Pan, H. Han, S. Shan, and X. Chen. Mean-variance loss for deep age estimation from a face. In *CVPR*, 2018. 1

[39] D. Ramanan and X. Zhu. Face detection, pose estimation, and landmark localization in the wild. In *CVPR*, 2012. 1

[40] A. Ranjan, T. Bolkart, S. Sanyal, and M. J. Black. Generating 3d faces using convolutional mesh autoencoders. In *ECCV*, 2018. 2, 3

[41] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016. 1

[42] J. Redmon and A. Farhadi. Yolo9000: better, faster, stronger. In *CVPR*, 2017. 1

[43] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 1, 2

[44] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic. 300 faces in-the-wild challenge: The first facial landmark localization challenge. In *ICCV Workshop*, 2013. 4

[45] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015. 1

[46] S. Sengupta, J.-C. Chen, C. Castillo, V. M. Patel, R. Chellappa, and D. W. Jacobs. Frontal to profile face verification in the wild. In *WACV*, 2016. 6

[47] A. Shrivastava, A. Gupta, and R. Girshick. Training region-based object detectors with online hard example mining. In *CVPR*, 2016. 2, 5

[48] B. M. Smith, L. Zhang, J. Brandt, Z. Lin, and J. Yang. Exemplar-based face parsing. In *CVPR*, 2013. 1

[49] X. Tang, D. K. Du, Z. He, and J. Liu. Pyramidbox: A context-assisted single shot face detector. In *ECCV*, 2018. 1, 2, 4, 5

[50] W. Tian, Z. Wang, H. Shen, W. Deng, B. Chen, and X. Zhang. Learning better features for face detection with feature fusion and segmentation supervision. *arXiv:1811.08557*, 2018. 5

[51] L. Tran and X. Liu. Nonlinear 3d face morphable model. In *CVPR*, 2018. 2

[52] L. Tran and X. Liu. On learning 3d face morphable model from in-the-wild images. *arXiv:1808.09560*, 2018. 2

[53] P. Viola and M. J. Jones. Robust real-time face detection. *IJCV*, 2004. 1, 2

[54] H. Wang, Z. Li, X. Ji, and Y. Wang. Face r-cnn. *arXiv:1706.01061*, 2017. 5

[55] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu. Cosface: Large margin cosine loss for deep face recognition. In *CVPR*, 2018. 1

[56] J. Wang, Y. Yuan, and G. Yu. Face attention network: an effective face detector for the occluded faces. *arXiv:1711.07246*, 2017. 2, 4, 5, 6

[57] Y. Wang, X. Ji, Z. Zhou, H. Wang, and Z. Li. Detecting faces using region-based fully convolutional networks. *arXiv:1709.05256*, 2017. 5

[58] B. Yang, J. Yan, Z. Lei, and S. Z. Li. Aggregate channel features for multi-view face detection. In *ICB*, 2014. 5

[59] S. Yang, P. Luo, C.-C. Loy, and X. Tang. From facial parts responses to face detection: A deep learning approach. In *ICCV*, 2015. 5

[60] S. Yang, P. Luo, C.-C. Loy, and X. Tang. Wider face: A face detection benchmark. In *CVPR*, 2016. 2, 3, 5

[61] S. Yang, Y. Xiong, C. C. Loy, and X. Tang. Face detection through scale-friendly deep convolutional networks. *arXiv:1706.02863*, 2017. 5

[62] S. Zafeiriou, C. Zhang, and Z. Zhang. A survey on face detection in the wild: past, present and future. *CVIU*, 2015. 1

[63] C. Zhang, X. Xu, and D. Tu. Face detection using improved faster rcnn. *arXiv:1802.02142*, 2018. 1, 2, 5

[64] F. Zhang, T. Zhang, Q. Mao, and C. Xu. Joint pose and expression modeling for facial expression recognition. In *CVPR*, 2018. 1

[65] J. Zhang, X. Wu, J. Zhu, and S. C. Hoi. Feature agglomeration networks for single stage face detection. *arXiv:1712.00721*, 2017. 5

[66] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *SPL*, 2016. 1, 2, 5, 6

[67] S. Zhang, R. Zhu, X. Wang, H. Shi, T. Fu, S. Wang, and T. Mei. Improved selective refinement network for face detection. *arXiv:1901.06651*, 2019. 2, 5, 6

[68] S. Zhang, X. Zhu, Z. Lei, H. Shi, X. Wang, and S. Z. Li. S3fd: Single shot scale-invariant face detector. In *ICCV*, 2017. 1, 2, 5

[69] Y. Zhang, X. X. Xu, and X. Liu. Robust and high performance face detector. *arXiv:1901.02350*, 2019. 5

[70] Y. Zhou, J. Deng, I. Kotsia, and S. Zafeiriou. Dense 3d face decoding over 2500fps: Joint texture and shape convolutional mesh decoders. In *arxiv*, 2019. 2, 3, 4, 6, 8

[71] C. Zhu, R. Tao, K. Luu, and M. Savvides. Seeing small faces from robust anchor's perspective. In *CVPR*, 2018. 5

[72] C. Zhu, Y. Zheng, K. Luu, and M. Savvides. Cms-rcnn: contextual multi-scale region-based cnn for unconstrained face detection. In *Deep Learning for Biometrics*. 2017. 2, 5

[73] S. Zhu, C. Li, C.-C. Loy, and X. Tang. Unconstrained face alignment via cascaded compositional learning. In *CVPR*, 2016. 6

[74] X. Zhu, H. Hu, S. Lin, and J. Dai. Deformable convnets v2: More deformable, better results. *arXiv:1811.11168*, 2018. 2, 4

[75] X. Zhu, Z. Lei, X. Liu, H. Shi, and S. Z. Li. Face alignment across large poses: A 3d solution. In *CVPR*, 2016. 6

[76] C. L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*, 2014. 3