Real-time Face Detection and Facial Landmark

Keyuan Zhang

Abstract

Face detectors and facial landmarks are fundamental to other advanced tasks, such as face recognition, liveness detection, and face editing. This paper will show how to integrate the two open-source works to build a high-performance facial landmark detector, the improvements and modifications I have made for each work. The integrated model can achieve over 24FPS on iPhone X.

1 Introduction

Nowadays, face-related tasks have commonly been applied in daily life, such as face recognition, liveness detection, face-related super-resolution, and face editing. These tasks require face detection and facial landmark as prior knowledge. Especially, face beauty is the requisite of some customers in real-time video communication. With the highly developed deep learning in computer vision, there are increasing deep learning based face-related works. To commercialize deep learning related works, in addition to accuracy, inference speed has become a metric to measure the practicability.

After an extensive literature review, I found many excellent works in face detection and facial landmark mainly focus on improving test accuracy but ignore the inference speed, which makes these models hardly be applied in real life. With the face beauty and face-focused super-resolution during video communication as the main objective, I modified, enhanced, and integrated two excellent works [2, 5] to achieve satisfactory accuracy while keeping the high efficiency.
To summaries, my main contributions are:

- Removed tiny face detection branch in RetinaFace-MobileNet-0.25 [2] and tuned the parameters in anchor boxes setting to increase inference speed about 30%.

At the same time, the detection accuracy of the normal scale face has almost not been affected.

- Modified the third-party open-source work [8] of PFLD [5] to improve the landmark detection accuracy, and replaced the loss function in PFLD with wing loss[4] to accelerate training convergence, and improve the overall landmark detection accuracy. Found the facial parts like eyebrow and contour that are hard to train after detailed analyzing, boosted the accuracy of these parts while maintaining the high performance of other parts through penalizing more in training.
- Augmented training dataset of RetinaFace-MobileNet-0.25 by adding random rotation$[-90^{\circ}, 90^{\circ}]$ to enhance the model robustness. Enhanced integration result by adding the inference result of RetinaFace-MobileNet-0.25 to the PFLD training dataset, which can increase the robustness of the integrated model as well.
- Modified model conversion tools [1] source code to solve the issues during converting.

2 Related Work

2.1 face detection

Single-stage v.s. two-stage: Two-stage [14, 4, 10] models divide face detection into two stages: the first stage output proposal regions, and the second stage classify and refine these regions. In general, the two-stage method achieves high location accuracy but sacrifices the detection speed. Therefore, two-stage models hardly reach real-time detection. Meanwhile, there are increasing works focus on single-stage detection to achieve high detection speed without compromising performance. single-stage [2, 9, 7, 6, 17] models output the final location coordinate and class probability without generate proposal regions firstly. Compared with two-stage methods, the single-stage detections are more efficient and have a higher recall rate but at the risk of achieving higher false positives and compromising the localization accuracy. Therefore, there are increasing works [2, 6] focused on single-stage detection try to improve detection accuracy while keeping efficiency.

2.2 facial landmark

Heatmap regression model v.s. coordinate regression model: Facial landmark detection is a regression task, the difficulty of the task can be influenced by the number of annotation points, and the quality of annotations. Commonly used dataset 300-W [11] and WFLW [15]are 68 and 98 points annotation, to achieve high performance on the challenge dataset, a heatmap is added to assist detection. Compared with the coordinate regression models, which are modeled without heatmap, the heatmap regression models have higher accuracy and are more stable in video facial landmark detection. However, the additional heatmap branch and the heatmap input increase the computation costs.

Global v.s. local: Due to the different detection difficulty of facial parts, the global to local models, like Coarse-toFine CNN, [18] predict the inner points and contour points separately, then separate the inner points into different parts (eg. eyebrows, eyes, nose, etc) and refine these parts. The global to local methods enhance the accuracy of difficult parts like contour and eyebrows but compromise the detection speed.

3 Methods and Improvements

3.1 Face Detection

The emphasis of the literature review has been focused on one-stage methods. I try to find a base model with excellent detection accuracy while maintaining high detection speed due to this task's objective. RetinaFace [2] adopts a multi-task learning strategy to simultaneously predict face score, face box, five facial landmarks, and 3D position and correspondence of each facial pixel. I will introduce the key design points of RetinaFace to provide essential background information in the following improvement work.

3.1.1 Model Architecture

The overall model can be divided into two parts: (1) encoder-decoder (2) context module. Encoder-decoder extracts semantic information while keeping image features during upsampling, and the context module captures tiny faces by implementing multiple 3×3 convolution layers to enlarge receptive field.

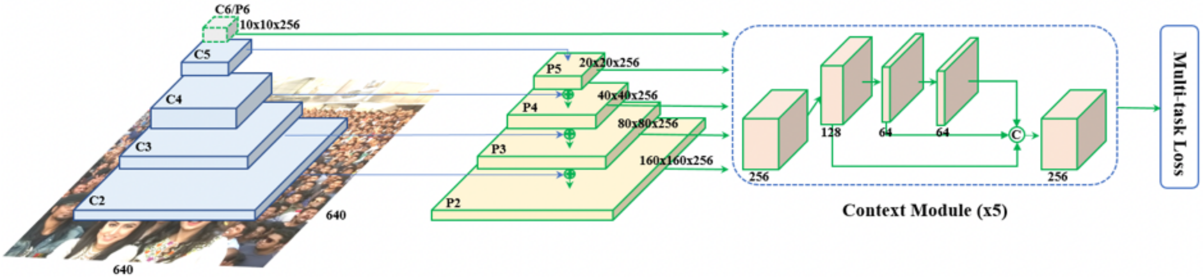


Figure 1: RetinaFace Model Architecture

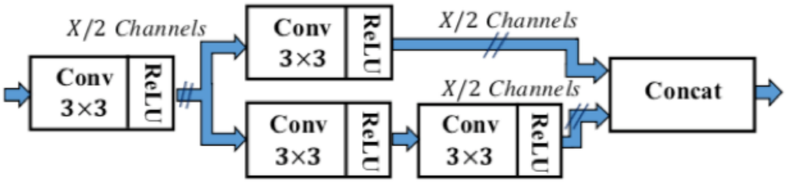


Figure 2: SSH context module

The anchor box design employs multi-scale settings to ensure multi-scale faces can be captured by the model, which corresponds to the feature pyramid encoder-decoder. Specifically, the anchor settings are scale-specific, for example, the stride-32 branch corresponds to 32 × 32 receptive filed in the input image to detect the large or medium faces, and the stride-4 branch is designed to capture tiny faces by tilling small anchors at the cost of more computational time and at the risk of more false positives.

Feature Pyramid	Stride	Anchor
P2 (160 × 160 × 256)	4	16, 20.16, 25.40
P3 (80 × 80 × 256)	8	32, 40.32, 50.80
P4 (40 × 40 × 256)	16	64, 80.63, 101.59
P5 (20 × 20 × 256)	32	128, 161.26, 203.19
P6 (10 × 10 × 256)	64	256, 322.54, 406.37

Table 1: The details of the 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 P2.

3.1.2 Loss Function

The loss function is the combination of multiple loss functions to satisfy the multi-task objective, which is defined as:

$$L = L_1(p_i, p_i^*) + \lambda_1 p_i^* L_2(t_i, t_i^*) + \lambda_2 p_i^* L_3(l_i, l_i^*)$$

L_1 applied softmax loss to monitor the face score (aka. face class probability).

L_2 is adopted here to supervise the training of face bounding box coordinates, 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.

L_3 helps the training of five landmarks, where $l_i = \{l_{x1}, l_{y1}, ..., l_{x5}, l_{y5}\}_i$ and $l_i^* = \{l_{x1}^*, l_{y1}^*, ..., l_{x5}^*, l_{y5}^*\}_i$ represent the predicted five facial landmarks and ground-truth associated with the positive anchor.

The bounding box coordinates and landmarks are ignored in negative anchors, therefore, if the face score is less than 0.5, the L_2 and L_3 are zero in this anchor.

3.1.3 Improvements

RetinaFace provides different backbones for different usages, mobilenet-0.25 backbone model is designed for highly efficient face localization. With the open-source work [2], I reproduced the result in the paper, but the inference speed is not satisfactory in the mobile device. It cost about 50ms for 640×640 input image, which considers as a high latency in real-time video communication.

Decreased the size of input image: To decrease detection costs dramatically, I first rescaled input image to 320×320, and it will bring about 75% speed improvement theoretically. However, the inference time was decreased only 50%. Without the time-consuming analysis of each layer, there are difficulties to find the large time-consuming layers and the time-consuming relationship between these layers and resolution, which means if future optimization is required in iOS, the detailed time-consuming analysis will be needed.

Removed stride-8 branch: RetinaFace-MobileNet-0.25 has three detection branches in total: stride-32, stride-16, and stride-8. Stride-8 branch is designed to detect the tiny faces in face detection challenge. Because the objective of this task is to build a face-related model that can meet the prerequisites of the objective, I removed the stride-8 branch in inference. It is worth mentioning that the removal of this branch may increase the training difficulty.

After applying the above methods, I reduced the inference time to 16ms/frame. Nevertheless, the modified model's accuracy loss is mainly due to the large drop of input resolution. Therefore, I retrained the model with 320 × 320 input image and tuned the parameters of anchor settings so that the stride-8 can be safely removed.

3.2 Facial Landmark

I chose PFLD [5] as the base model for the reason of efficiency and performance. The main objective of PFLD is to build a real-time facial landmark detection model and the model performances well in the 300-w dataset. Just as in the last section, I will introduce the key design points of PFLD to provide essential background information in the following improvement works.

3.2.1 Model Architecture

Backbone Network: The overall structure is compact, and the heatmap or global-to local refinement is unused due to the request of real-time inference. Mobilenet backbone is adopted in PFLD, and the fully connected layer is applied in the last to output landmarks. The auxiliary network is added to predict the rotation information, which helps the convergence of training and makes the landmark localization stable and robust. It is worth mentioning that the auxiliary network applies only in training, and it is removed in inference to reduce computation costs. The model architecture is shown in Figure 3.

Auxiliary network: Due to the large scale of the difference between prediction points and ground truth, processed training loss indiscriminately makes the training process hard to convergent and impairs the robustness. The auxiliary network in PFLD ameliorates this issue.

For every input data, the auxiliary network predicts 3D Euler angles that are pitch, yaw, and roll, and the ground truth is computed from ground truth landmarks. For accuracy, the angles may not be exact for each face, but they are sufficient to classify datasets. It should be noticed that the input of the auxiliary net is the 4-th block of the main net, not the input image.

3.2.2 Loss Function

The loss function is the novel part of PFLD, and the loss function can be divided into three parts based on the model structure. The first part is used to compute the

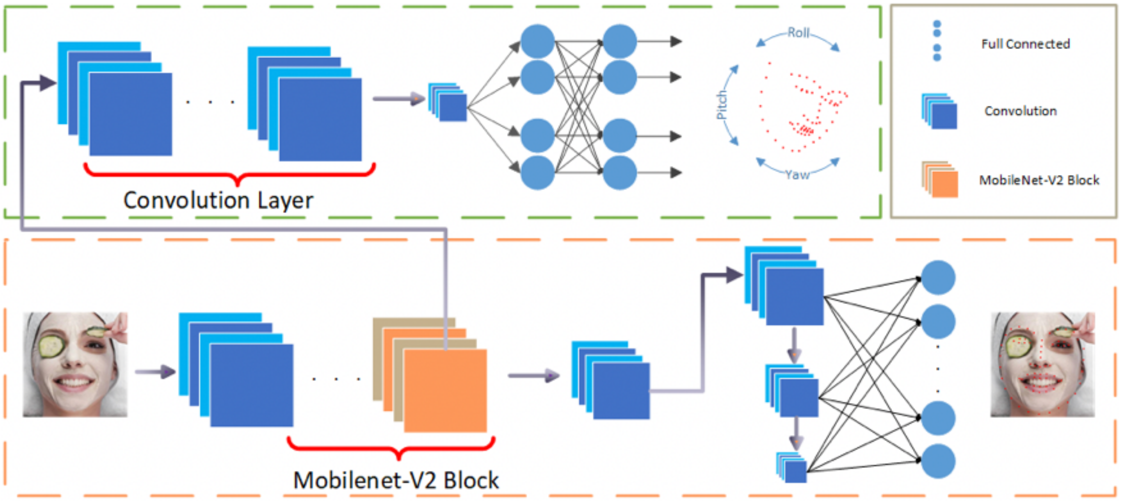


Figure 3: PFLD Model Architecture

L2 loss between the prediction and ground truth. The second part designs to monitor the loss of predicted Euler angles and the last one penalizes each class differently to ameliorate the sample imbalance. The loss function is defined as:

$$L := \frac{1}{M}\sum_{m=1}^{M}\sum_{n=1}^{N}\left(\sum_{c=1}^{C}\omega_n^c\sum_{k=1}^{k}(1 - cos\theta_n^k)||d_n^m||_2^2\right)$$

In which, $\theta 1$, $\theta 2$, and $\theta 3$ (K=3) represent the angles of deviation between the ground-truth and estimated yaw, pitch, and roll angles. The weighting parameter ω_n^c is adjusted according to the fraction of samples belonging to class c (this work simply adopts the reciprocal of fraction).
Without θ and ω, the loss function is a simple L2 loss.

3.2.3 Improvements

PFLD is not an open-source project, I found an excellent third-party [8] open-source work in GitHub that reproduce the most of implementation details. However, the undisclosed details bring a lot of troubles in implementation. Thus the improvements have to be made to utilize this project, and I will introduce the major improvements in the following section.

Rotation without cropping: Prepossessing of input plays important role in model training, however, this part has not been disclosed in the paper in detail. And I found the original rotation method in this work [8] that is rotation with cropping impairs the accuracy significantly. This is understandable because this operation brings loss of image information at the beginning and undermines the performance in the common set. Therefore, I replace this operation with rotation without cropping.

Wing loss function: I realized that the training process convergent very slow even though the auxiliary net is adopted. This is because the design of the loss function focuses more on the minority of hard samples, which impeding the overall training. After a literature review of face-related loss functions, I found wing loss [3] helps model with convergent. The wing loss is defined as:

$$wing(x) = \begin{cases} \omega \ln(1 + |x|/\epsilon), & \text{if } |x| < \omega \\ |x| - C, & \text{otherwise} \end{cases}$$

x is the absolute difference between prediction and ground truth, total loss grows when $x < \omega$, vice-versa. So the large scale of difficulty on each facial part landmarks will not affects the training significantly.

Besides, the eyebrow and contour have lower accuracy after a detailed analysis of facial parts. To improve the accuracy of these parts, I tuned the loss weight to $1.15 : 1$

Integration the integrated facial landmark detection model should include the face detection model firstly, which means the output of the face detection model is the input of the landmark detection model. To better achieve this, I augmented the input dataset of PFLD-v2* with the prepossessed dataset of the output of RM-0.25-v2-w8* and retrained the model.

4 Evaluation

4.1 Face Detection

For detection, I applied the standard evaluation metric of average precision(AP) when $IoU = 0.5$ on the test dataset, and feed the output file of predicted bounding box into the evaluation script to get AP based on the requirements of WIDER FACE [16].

WIDER FACE divides the test dataset into Easy, Medium, and Hard subsets based on the detection difficulty. However, this separation helps a little on my objective because face beauty and face-focused super-resolution rarely consider tiny faces, extreme occlusions, and lighting. Therefore, I reorganized the dataset into Small,

Method	Smallface	Mediumface	Largeface
Average Precision (AP)			
RM-0.25	0.681	0.908	0.962
RM-0.25-w8	0.355	0.904	0.973
RM-0.25*	0.378	0.838	0.956
RM-0.25-w8*	0.014	0.477	0.964
RM-0.25-v2*	0.500	0.828	0.950
RM-0.25-v2-w8*	0.254	0.836	0.964

Table 2: Comparison of modified models. The row marked with displays the result using 320 × 320 input size. v2 is the modified model, and w8 means the model without stride-8 branch

Medium, and Large subsets based on the size ratio of the faces to the whole picture and filtered out 39% of the test dataset.

The truncation of stride-8 affects a little on Large and Medium subsets, but the Small one is based on Table 1. Therefore, I can safely remove this stride to decrease inference latency. Besides, down-sampling input size and tunning anchor settings without changing modifying strategy and datasets affect Medium subsets' accuracy to some extent, due to lowering input size brings difficulty to training and impairs convergence. The possible solution will be discussed in future work.

4.2 Facial Landmark

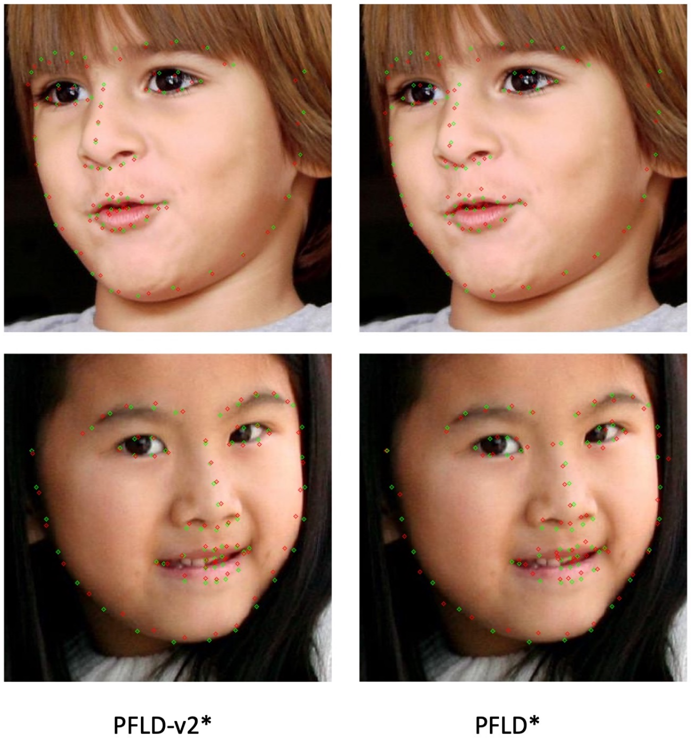
The overall accuracy has improved a lot from the [8], but some challenging facial parts like contour and eyebrows still has much room for improvement. This is because these are outer parts and are easily affected by the detection bounding box. The possible solution will be discussed in future work.

Method	Common	Challenge	Full
Inter-ocular Normalization (ION)			
PFLD*	5.18	8.78	5.85
PFLD-v2*	4.19	7.68	4.87

Table 3: Comparison of modified models. The row marked with displays the result using third-party reproduction. v2 is the modified model

Method	Contour	Eyebrow	Eyes	Nose	Mouth
Inter-ocular Normalization (ION)					
PFLD*	8.46	5.56	4.33	5.40	5.01
PFLD-v2*	7.26	5.33	3.24	3.54	4.20

Table 4: Comparison of modified models in facial parts. The row marked with displays the result using third-party reproduction. v2 is the modified model



PFLD-v2*

PFLD*

Figure 4: Visual Improvement of PFLD-v2*. Green points denote ground truth, red points denote prediction.

5 Application

To test the performance of the integrated model, I developed an iOS demo with the CoreML mobile inference framework. However, I encountered some tricky issues during model conversion. The switch [12] and merge [13] ops in TensorFlow batch-norm layer are not well supported in CoreML, which means we cannot solve this issue by modifying conversion tools. According to the definition, the two ops are ignored in inference, therefore, I can safely skip these two layers by modifying the inputs and outputs of the related ops.

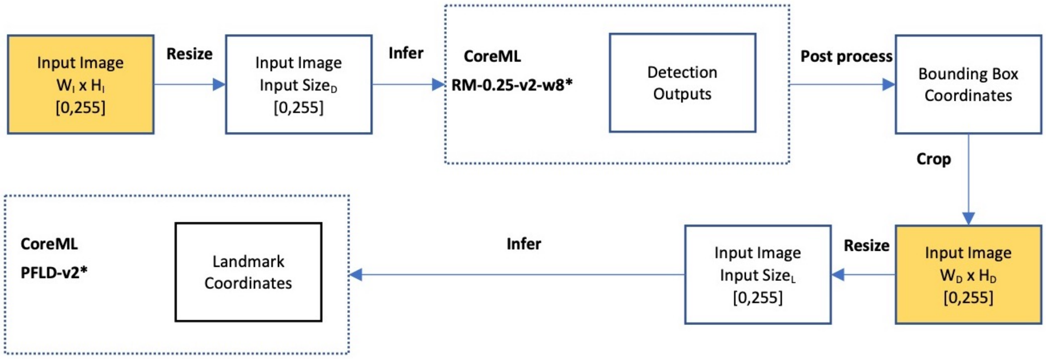


Figure 5: iOS Demo Flowchart

6 Future Work

Improvements on the mediumface subset: Because lowering input size brings difficulty to training and impairs convergence, adjust anchor settings based on the downsampling ratio are less helpful. To enhance the accuracy of the mediumface subset, I need to reclassify the training dataset and filter out the smallface subset.

Improvements on eyebrows and contour localization: The eyebrows and contour are the outer parts of the face and are easily affected by the detection bounding box. Therefore, the possible solutions are: (1) add boundary heatmap [15] to assist outer parts regression (2) design an experiment to find effective bounding box settings.

A new evaluation metric: Finding an effective bounding box setting needs a more comprehensive metric to evaluate the location accuracy of the predicted bounding box. Therefore, a new evaluation metric of localization accuracy will be needed.

7 Conclusion

I demonstrated an integrated face detection and landmarks model that aims for face beauty and face-focused super-resolution in this work. The proposed model is modified from the two excellent open-source works by extensive literature review and experiments that achieve high efficiency and accuracy. Moreover, the proposed model is the base model of face-focused video enhancement, like face beauty, and works to improve real-time video communication efficiency if we applied for this work in face-focused super-resolution. Finally, I mentioned a few improving thoughts in future work.

References

[1] Apple. https://github.com/apple/coremltools. Accessed: 2020-04-30. 1

[2] J. Deng, J. Guo, Y. Zhou, J. Yu, I. Kotsia, and S. Zafeiriou. Retinaface: Single-stage dense face localisation in the wild. *arXiv preprint arXiv:1905.00641*, 2019. 1, 2.1, 3.1, 3.1.3

[3] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2235–2245, 2018. 3.2.3

[4] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 2.1

[5] X. Guo, S. Li, J. Yu, J. Zhang, J. Ma, L. Ma, W. Liu, and H. Ling. Pfld: A practical facial landmark detector. *arXiv preprint arXiv:1902.10859*, 2019. 1, 3.2

[6] Y. He, D. Xu, L. Wu, M. Jian, S. Xiang, and C. Pan. Lffd: A light and fast face detector for edge devices. *arXiv preprint arXiv:1904.10633*, 2019. 2.1

[7] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016. 2.1

[8] G. QI. https://github.com/guoqiangqi/PFLD. Accessed: 2020-04-30. 1, 3.2.3, 4.2

[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 2.1

[10] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016. 2.1

[11] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic. 300 faces in-the-wild challenge: The first facial landmark localization challenge. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 397–403, 2013. 2.2

[12] TensorFlow. `https://www.tensorflow.org/api_docs/cc/class/tensorflow/ops/switch`. Accessed: 2020-04-30. 5

[13] TensorFlow. `https://www.tensorflow.org/api_docs/python/tf/compat/v1/summary/merge`. Accessed: 2020-04-30. 5

[14] H. Wang, Z. Li, X. Ji, and Y. Wang. Face r-cnn. *arXiv preprint arXiv:1706.01061*, 2017. 2.1

[15] W. Wu, C. Qian, S. Yang, Q. Wang, Y. Cai, and Q. Zhou. Look at boundary: A boundary-aware face alignment algorithm. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2129–2138, 2018. 2.2, 6

[16] S. Yang, P. Luo, C.-C. Loy, and X. Tang. Wider face: A face detection benchmark. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5525–5533, 2016. 4.1

[17] S. Zhang, X. Zhu, Z. Lei, H. Shi, X. Wang, and S. Z. Li. Faceboxes: A cpu real-time face detector with high accuracy. In *2017 IEEE International Joint Conference on Biometrics (IJCB)*, pages 1–9. IEEE, 2017. 2.1

[18] E. Zhou, H. Fan, Z. Cao, Y. Jiang, and Q. Yin. Extensive facial landmark localization with coarse-to-fine convolutional network cascade. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 386–391, 2013. 2.2