future internet

MDPI

Article

A Fast and Lightweight Method with Feature Fusion and Multi-Context for Face Detection

Lei Zhang [1] **and Xiaoli Zhi** [1,2,*]

[1] School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China; zzzzzl@shu.edu.cn

[2] Shanghai Institute for Advanced Communication and Data Science, Shanghai University, Shanghai 200444, China

* Correspondence: xlzhi@shu.edu.cn; Tel.: +86-139-1848-3274

Received: 23 July 2018; Accepted: 14 August 2018; Published: 17 August 2018

Abstract: Convolutional neural networks (CNN for short) have made great progress in face detection. They mostly take computation intensive networks as the backbone in order to obtain high precision, and they cannot get a good detection speed without the support of high-performance GPUs (Graphics Processing Units). This limits CNN-based face detection algorithms in real applications, especially in some speed dependent ones. To alleviate this problem, we propose a lightweight face detector in this paper, which takes a fast residual network as backbone. Our method can run fast even on cheap and ordinary GPUs. To guarantee its detection precision, multi-scale features and multi-context are fully exploited in efficient ways. Specifically, feature fusion is used to obtain semantic strongly multi-scale features firstly. Then multi-context including both local and global context is added to these multi-scale features without extra computational burden. The local context is added through a depthwise separable convolution based approach, and the global context by a simple global average pooling way. Experimental results show that our method can run at about 110 fps on VGA (Video Graphics Array)-resolution images, while still maintaining competitive precision on WIDER FACE and FDDB (Face Detection Data Set and Benchmark) datasets as compared with its state-of-the-art counterparts.

Keywords: convolutional neural networks; face detection; feature fusion; context; speed; precision

1. Introduction

Face detection is a key step in many visual applications, such as face verification, face tracking and etc. In recent years, convolutional neural network (CNN)-based general object and face detection methods [1–6] have achieved great success. To obtain better accuracy, these methods mainly take computation intensive convolutional networks such as VGG-16 [7] or ResNet (Residual Network)-101 [8] as backbone. Although these networks are very powerful, their big computation workload causes poor detection speed and thus constrains their applicability in real life. For example, video applications require a detection algorithm to process at least 30 video frames per second and methods [9,10] based on traditional machine learning are still heavily used. Though there are some recent real time methods [6,11], they can only obtain this fast speed by the support of high performance GPUs. These GPUs will be expensive and could be a financial burden for users. Nvidia has introduced a Pascal-powered Jetson TX2 computer for real life applications, but its computing power is 1.5 TFLOPs compared to Titan X's 11 TFLOPs, which is still too weak to support recent CNN-based methods without much accuracy loss. To address this issue, we propose a fast and lightweight face detector. It takes a fast residual network as backbone to make itself 'lightweight'. Our method can easily reach a high inference speed even on a cheap and ordinary GPU. To obtain a detection precision that could be comparable to the computation intensive convolutional network based methods, we employ multi-scale features and multi-context through some efficient ways.

Multi-scale features are crucial to detect objects in various scales [5,6,12]. Image pyramid is a widely used approach for getting multi-scale features [5]. However, it requires large storage and big calculation thus resulting in very low speed. A more efficient way to get multi-scale features is to leverage the inherently hierarchical features within a CNN [6,12]. In CNN, after several pooling operations, outputs of convolutional layers have different scales at different depths. The lower layers are of higher resolution and upper layers are much coarser. These outputs can be used as multi-scale features to detect different scales of faces without extra computational cost. We predict faces from multiple layers of the network, and take a feature fusion approach to fuse high level semantic features with low level but high-resolution features.

Contextual information is another important factor for detection. Previous studies [5,13–15] indicated that objects would be much easier to be recognized with context added. The existing approaches described in the literature are not suitable for our method, as we want to add context with no additional computational burden. In this paper, multi-context including both local and global contexts are exploited. Local context means context incorporated from some local regions of the feature maps, and global context means the context incorporated from the whole feature maps. We propose a depthwise separable convolution [16] based approach to add local context to the fused feature maps, and a simple pooling way to add global context.

After feature fusion and adding context, the faces will be predicted in a single stage. Similar to RPN (Region Proposal Network) [17], each scale of the feature maps is used to regress a set of predefined anchors towards faces. But the final classification takes place together with the regression, which differs from RPN. We integrate the predictions of each scale and obtain final best results through non-maximum suppression (NMS for short).

2. Related Works

2.1. Face Detection

Recently, CNN-based face detection methods have significantly surpassed traditional methods such as HOG (Histogram of Oriented Gradient) [18] and VJ (Viola-Jones) [19]. Cascade-CNN [20] developed a cascaded architecture on CNNs with powerful discriminative capability and high performance. Faceness [21] demonstrated that CNN models with facial attributes can be applied to find face proposals and these proposals can be further processed by an AlexNet-like CNN. CMS-RCNN [15] added body contextual information that surrounded the face in Faster RCNN. SSH [22] predicted faces from multiple layers and added contextual information to increase accuracy. Our method differs from SSH in that a top-down path is used to refine the low-level features with deep semantically strong features and not only local but also global context is added.

2.2. Multi-Scale Features and Context

Multi-scale features are widely exploited to handle scale variance in object detection. SSD [6] made predictions from multiple convolutional layers of VGG-16 [7] and got an improvement on detecting multi-scale objects. FCN [12] used skip connections to add features from shallow layers. FPN [23] constructed feature pyramids by using bottom-up, top-down and lateral connection to fuse shallow but high-resolution features with deep but semantically strong features.

Context is another crucial factor for multi-scale object detection, especially for small objects. MultiPath [14] enlarged the bounding box to cover larger area surrounding the object, then it used a pooling layer to extract features from the enlarged bounding box. HR [5] used different size templates to add contextual information around faces to facilitate detecting small faces. ION [13] used a powerful stacked IRNN module to add global context and achieved a better performance over previous methods.

2.3. Context Adding Approaches

In two-stage object detection methods, a common way to add context is to enlarge the window around the target, as shown in Figure 1a. Different sizes of enlarged windows will add multiple surrounding information. Each enlarged window will be processed by a RoI (Region of Interest)-pooling layer to extract feature for further prediction. But this approach increases computation and ParseNet [24] used a more efficient way which is incorporating the contextual pixels. As shown in Figure 1b, global pooling is used to incorporate the pixel information of the whole feature maps to produce global context. This global context is then combined with the original feature maps by concatenation for further prediction.

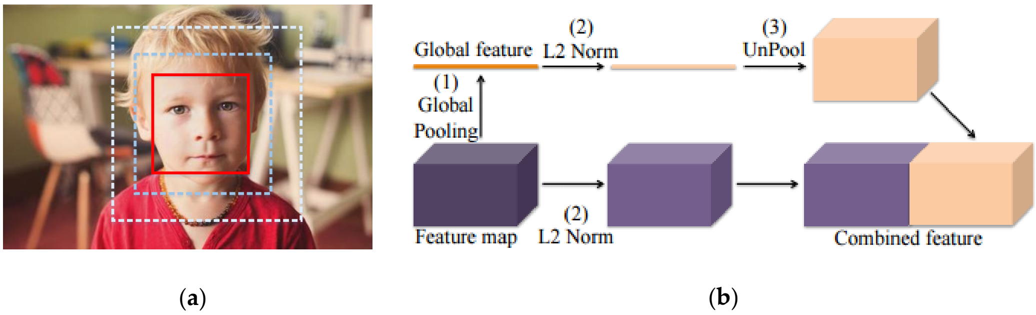


Figure 1. Two context adding approaches. (**a**) Red window is the original bounding box and the dashed windows are the enlarged windows; (**b**) Using global pooling to incorporate global pixel information.

In our method, context means the neighboring pixels around the target pixel, different sizes of neighboring pixels are incorporated to improve the detection ability. We introduce a depthwise separable convolution based approach to incorporate local context from each scale of the fused feature maps and then add back to it. Depthwise separable convolution is used to factorize the conventional convolution as it can help to substantially reduce computation and parameters. To add global context, global information is incorporated from the most semantically strong feature maps and added to each scale of the fused feature maps separately.

3. Methods

Our method takes a fast residual network ResNet-18 [8] as backbone to achieve fast face detection. This network takes advantage of a residual architecture, as shown in Figure 2. It only has 1.3 billion FLOPs comparing to VGG-16's 15.3 billion FLOPs, and it has a 71% top-1 accuracy and 89% top-5 accuracy on ImageNet [25]. This network only takes about 2 ms for inference on a Titan X GPU. Based on this network, we exploit multi-scale features and multi-context in order to improve detection precision for multi-scale faces.

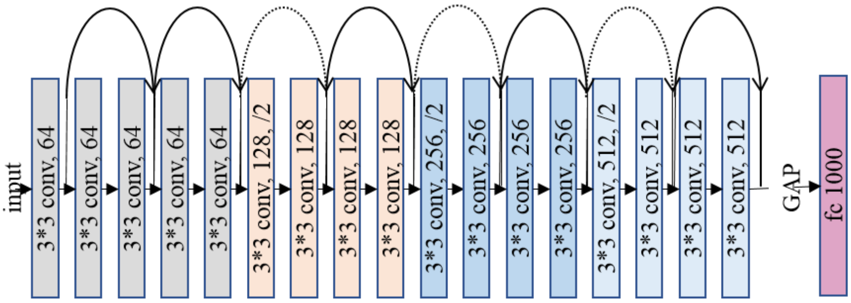


Figure 2. Structure of the ResNet-18. The dotted shortcuts increase dimensions.

3.1. Architecture

The overall architecture of our method is shown in Figure 3. During the inference process, each layer will produce a set of scale specific output feature maps; we make prediction on several sets of feature maps which are pre-selected from multiple layers. To enhance semantic features of the shallow layers, feature fusion [23,26] is used to fuse semantically strong features of deep layers to them. This procedure produces three sets of feature maps on three scales, and then context is added to these feature maps. Three local context paths are attached on each scale of feature maps to incorporate three sizes of pixel areas, i.e., three scales of context. And a global context path is used to incorporate global size of pixel areas of the feature maps, but this global context is only obtained from the last layer conv17. Feature maps of this layer are the most semantically strong, thus global information of these feature maps is the most representative. We obtain global context from conv17, rescale and concatenate (channel-wise) respectively with the local context incorporated from each scale of the fused feature maps. The concatenated context then merge with the original feature maps by element-wise concatenation. An approach similar to RPN is finally used to predict faces from each scale of the final feature maps simultaneously.

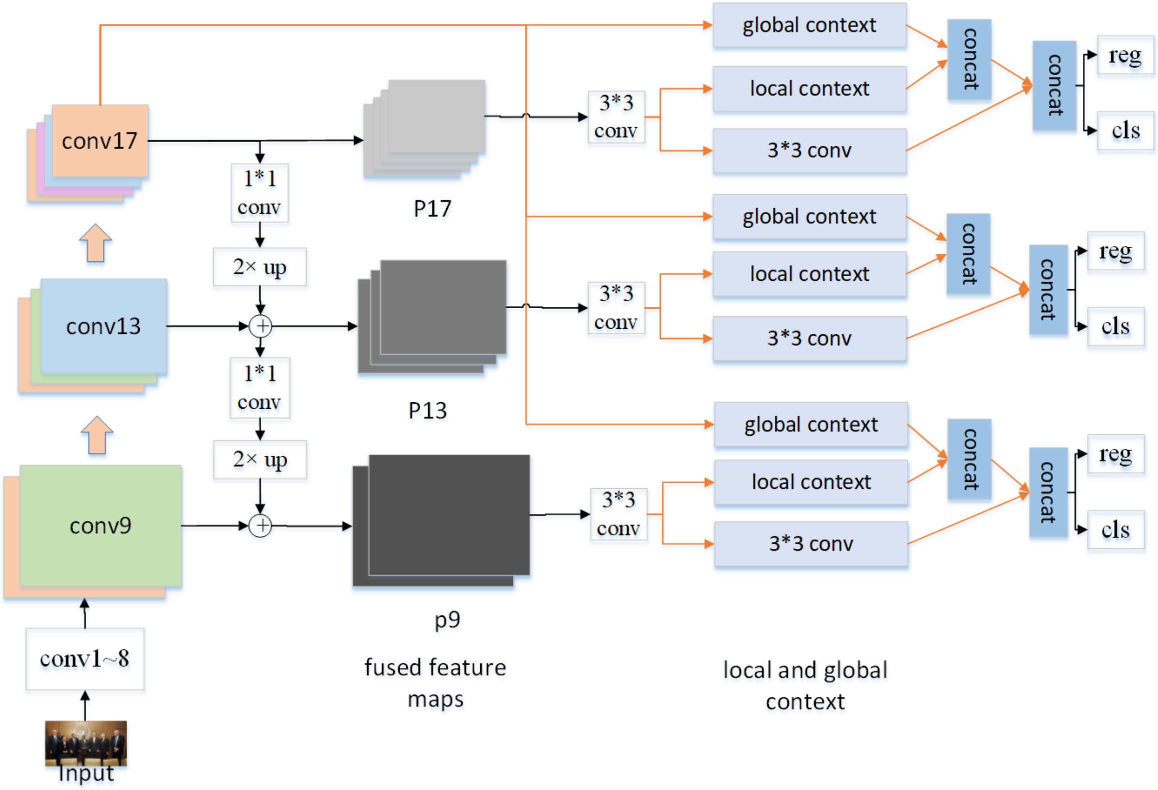


Figure 3. Overall architecture of our method.

3.1.1. Multi-Scale Features

As mentioned above, we use outputs of several selected convolutional layers to obtain multi-scale features. We first choose output feature maps with different scales produced from the forward process of the network. Using outputs before the 8th layer will bring large calculations, so we choose three sets of outputs which are produced by conv9, conv13 and conv17, with a scaling step of 2 and strides of 4, 8 and 16.

Since shallow layers are semantically weak and not suitable for prediction, we then fuse the deep semantically strong features to the shallow high-resolution feature maps by a top-down pathway, as illustrated in Figure 3. First, channels of output feature maps of conv17 are reduced to 256 through a 1 × 1 convolutional layer. Then, the reduced outputs are bilinearly up-sampled and merged with outputs of conv13 by element-wise summation. The merged feature maps are then merged with the outputs of conv9 in the same way but with channels of 128. After two iterations, we obtain two sets of fused feature maps and in total, three sets of feature maps with semantically strong features at all three scales, called P9, P13 and P17. A 3 × 3 convolutional layer is attached after each set of the feature maps with channels of 512, 256 and 128.

3.1.2. Local Context Gathering by Depthwise Separable Convolution

In addition to feature fusion, we also consider adding multi-context to improve the capability of detecting faces. In our method, several local context paths are attached on top of P9, P13 and P17 to add local contextual information and a global context path starting from conv17 is used to add global contextual information, which is inspired by [13,22,24,27].

In CNN, convolutional filters extract features from input feature maps by way of a sliding window. Different sizes of filters will extract different sizes of pixel areas, such as 3 × 3, 5 × 5 and etc., so these filters can naturally be used as context extractors. With these context extractors, we propose a depthwise separable convolution based approach to add local context. Depthwise separable convolution [16] is a factorization method that can greatly reduce computation while maintaining the capability of conventional convolutions. Unlike conventional convolution, in depthwise separable convolution, the 3 × 3 filters first convolve each channel of the input feature maps independently, and then 1 × 1 filters are used to compute the linear combination of the previous results. With depthwise separable convolution, calculation will be reduced by 8–9 times and our detection speed can then effectively be promoted.

We use three paths to add local context. Different sizes of neighboring pixels are incorporated together to add different scales of context. As illustrated in Figure 4, the first local context path uses two stacked 3 × 3 depthwise separable convolution to incorporate 5 × 5 size of pixels of the input feature maps, which adds 2 pixels worth of context to the original feature maps. The second local context path uses three stacked 3 × 3 depthwise separable convolution to incorporate 7 × 7 size of pixels of the input feature maps, which adds 4 pixels worth of context. And the third local context path incorporates 9 × 9 size of pixels, which adds 6 pixels worth of context. Therefore, our method will incorporate three scales of local context. A batch normalization layer and a ReLU nonlinearity layer are followed each convolutional layer. The number of input and output channels of each path are uniformly set to 128, 64 and 32 with respect to each scale of feature maps.

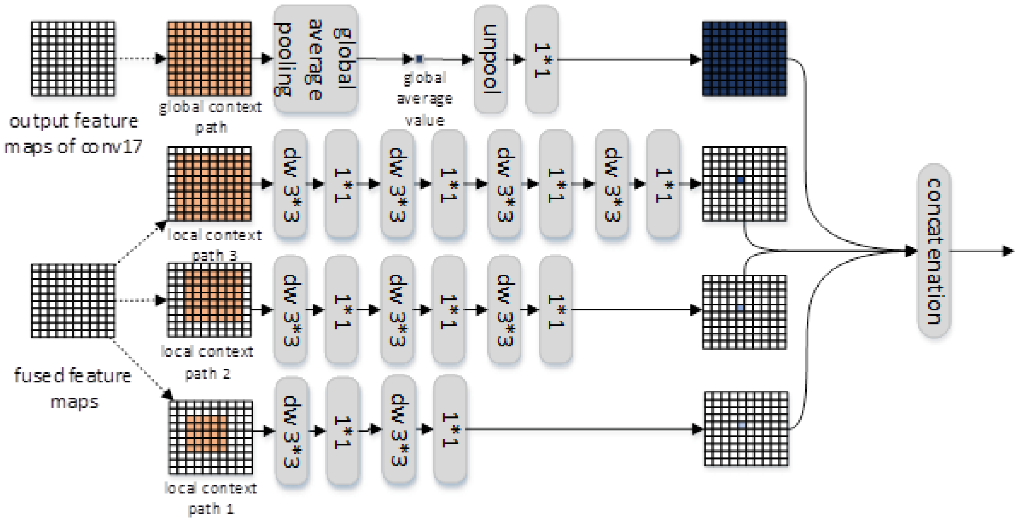


Figure 4. Context incorporation. Depthwise separable convolution (dw for short) is used to reduce computation, each convolutional layer is followed by a batchnorm layer and a ReLU layer.

3.1.3. Global Context Gathering by Global Average Pooling

We chose an approach called global average pooling to gather global context. This approach, successfully used to add global context in semantic segmentation tasks [24], is very conveniently deployed and it only consumes a few computations. As shown in Figure 4, the global context path first uses global average pooling to incorporate global context of conv17's output feature maps to figure out a global average value of the whole feature map. This value is then unpooled (repeated spatially) to the same spatial size of each scale of the fused feature maps, and we get three sets of global feature maps. A 1 × 1 convolutional layer is used to reduce the number of output channels to 128, 64 and 32 respectively. Then each set of the global feature maps is concatenated (channel-wise) with the local context maps separately. Then three sets of feature maps that contain local and global context are produced. Finally, these feature maps are added to the original fused feature maps by element-wise concatenation.

3.2. RPN-Like Approach to Predict Faces

A similar approach to RPN [17] is used to predict faces in our method. One 1 × 1 convolutional layer is used to classify the predefined anchors, but another 1 × 1 convolutional layer is used to regress the coordinates of anchors at the same time, which is the major difference from RPN. We make prediction on each scale of the final feature maps respectively. Then the predictions of each scale are integrated together. Finally, non-maximum-suppression (NMS) is used to obtain the best results. Only anchors with aspect ratio of 1 are used, and sizes of anchors are set to {16 × 16, 32 × 32}, {32 × 32, 64 × 64} and {64 × 64, 128 × 128} with respect to P9, P13 and P17.

3.3. Loss Function

We use the multitask loss function defined in RPN [17] to optimize the model parameters:

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i, t_i^*) \tag{1}$$

In Equation (1), i is the index of the anchor, and p_i is the predicted probability of the i-th anchor. If an anchor is positive, its confidence p_i^* is set to 1, otherwise p_i^* will be set to -1. t_i is a parameterized coordinate vector representing a predicted bounding box, and t_i^* is a parameterized coordinate vector of a ground truth bounding box related to the positive anchor. The classification loss function $L_{cls}(p_i, p_i^*)$ is a softmax loss function of two classes, and the regression loss function $L_{reg}(t_i, t_i^*)$ is a smooth L_1 loss function defined in Fast RCNN [3]. $p_i^* L_{reg}(t_i, t_i^*)$ indicates that the regression function can only be activated by the positive anchor. N_{cls} and N_{reg} are used to regularize the two indicators and the parameter λ is used to balance the two indicators. As mentioned in Section 3.2, as classification and regression take place together, we set λ = 1 in our method.

4. Experiments

4.1. Experimental Setup

WIDER FACE [28]: This dataset contains 32,203 images and 393,703 annotated faces. The dataset is split into training (40%), validation (10%) and testing (50%) set. Validation set and testing set are divided into 'easy', 'medium' and 'hard' subset. With its rich multi-scale and occluded faces, WIDER FACE is one of the most challenging public face datasets. We trained our model on the training set and evaluated it on the validation set.

FDDB (Face Detection Dataset and Benchmark) [29]: FDDB is one of the most popular face detection evaluation platforms in the world. It contains 2845 images with 5171 faces. When evaluating our method on FDDB, we convert the face annotation area to rectangle.

Training settings: We use weights pre-trained on ImageNet [25] to initialize the original classification network. Then, we fine-tune our network on the WIDER FACE training set. Stochastic gradient descent algorithm is used to train the network, batch size is set to 32. Learning rate is 0.0001 at the beginning and momentum is 0.9. After 80 k iterations, learning rate is reduced by 10 times and the total iterations are 100 k. We trained and evaluated our method on a Titan X GPU.

4.2. Results on WIDER FACE

We evaluate our method on WIDER FACE from four aspects.

4.2.1. Comparison with the State-of-the-Art Face Detection Methods

Evaluation results on WIDER FACE validation set are shown in Table 1. Our method's APs (AP is short for Average Precision) achieves 90.6%, 89.1% and 79.8% respectively on the three subsets. Our method outperforms most of the listed methods and catches up with the performance of some 'heavy' VGG-16 based methods. FD-CNN [30] is a recent lightweight method, and ours has much higher APs than FD-CNN on all three subsets. Though FD-CNN takes a shallow backbone, it uses image pyramid during testing phase and produces large extra computations. This makes its speed slower than ours. When compare to three VGG-16 based methods, our method surpasses CMS-RCNN [15] and obtains almost the same precision as that of recent MSCNN [31]. Though the AP of SSH [22] is better than ours, it takes many complex training approaches to improve accuracy and its VGG-16 backbone makes itself three times slower than ours. In other words, our method can run at a much faster speed while achieving a proximate precision, as compared with its popular counterparts.

Table 1. Comparison between our method and its state-of-the-art peers.

Methods	Inference Time (ms)	AP (Easy)	AP (Medium)	AP (Hard)
Two-stage CNN [20]	10	68.1	61.8	32.3
Multiscale Cascade CNN [25]	-	69.1	66.4	42.4
Faceness [21]	50	71.3	78.8	34.5
FD-CNN [30]	30	77.4	74.0	51.0
Multitask cascade CNN [32]	10	84.8	82.5	59.8
CMS-RCNN [15]-VGG-16	-	89.9	87.4	62.4
MSCNN [31]-VGG-16	25	91.6	90.3	80.2
SSH [22]-VGG-16	30	93.1	92.1	84.5
Ours-without feature fusion and context	6	81.2	80.2	71.6
Ours-without context	7	86.1	84.5	76.8
Ours	9	90.6	89.1	79.8

4.2.2. The Effectiveness of Our Multi-Scale Features and Multi-Context Design

To evaluate the effectiveness of our design, we first test our method without using multi-scale features and context (i.e., it predicts faces only from multiple layers). The APs are 81.2%, 80.2% and 71.6% respectively. We then evaluate our method without context but with feature fusion, the APs are 86.1%, 84.5% and 76.8% respectively. Compared to the high APs of 90.6%, 89.1% and 79.8% of our full version method, it can be concluded that both feature fusion and context are important for the AP. The 'hard' subset mainly contains faces with large scale changes, and our method still performs well on this subset. It implies our method is powerful for detecting multi-scale faces. Feature fusion can produce semantically strong features and the multi-context gathered from different sizes of neighboring pixels will bring more robust to scale changes; this extra information is also important for detecting small faces. This gives confidence to the effectiveness of our design, especially our context adding mechanism. On the other hand, the inference speeds of our three method variants are 6 ms, 7 ms and 9 ms, and it can be assumed that our design only brings little extra computation.

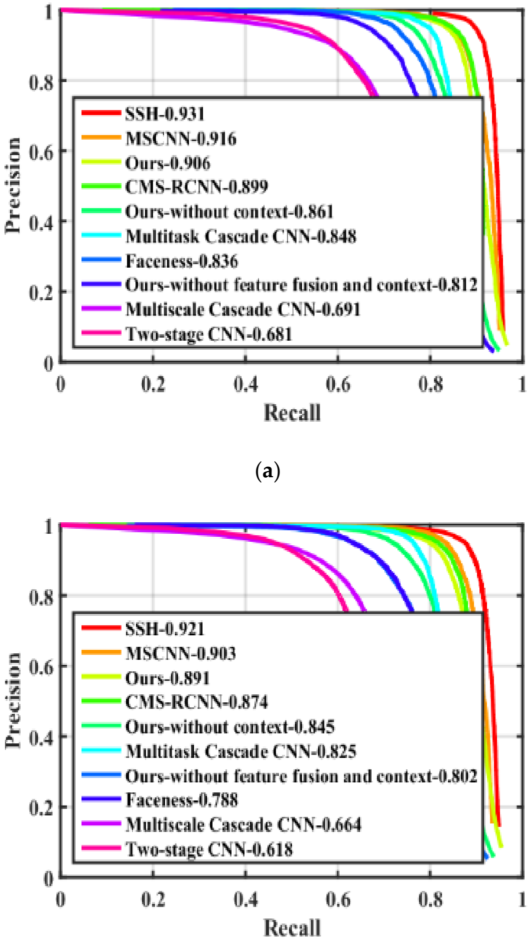
4.2.3. Computational Complexity

Standard convolutions have the computational cost of $D_k \times D_k \times M \times N \times D_F \times D_F$, where $D_k \times D_k$ is the kernel size, M is the number of input channels, N is the number of output channels and $D_F \times D_F$ is the feature map size. And in depthwise separable convolutions the cost is $D_k \times D_k \times M \times D_F \times D_F + M \times N \times D_F \times D_F$, which is the sum of the 3 × 3 depthwise convolutions and the following 1 × 1 convolutions. By depthwise separable convolution we get a reduction in computation of $\frac{1}{N} + \frac{1}{D_K^2}$. In our method, context incorporation only brings 1.46 billion FLOPs and the total computation cost is 3.2 billion FLOPs during training and testing phases. SSH and MS-CNN has approximately 17.7 billion FLOPs and 17.2 billion FLOPs respectively, their 'heavy' VGG-16 backbones bring most of the computation. As a result, our method is much computational efficiency comparing to SSH and MS-CNN.

4.2.4. Comparison of Inference Speed

To effectively assess the detection speed of our method, the image size is rescaled to the same 640 × 480 VGA resolution as that in other methods [20,21,31]. Inference time of our method is only 9 ms, which is the best result among similar methods. Although Two-stage CNN [20] and Multitask cascade CNN [31] have almost the same speed as ours, they perform much worse than ours on the three subsets of WIDER FACE. And our method is three times faster than recent VGG-16 based methods [22,32].

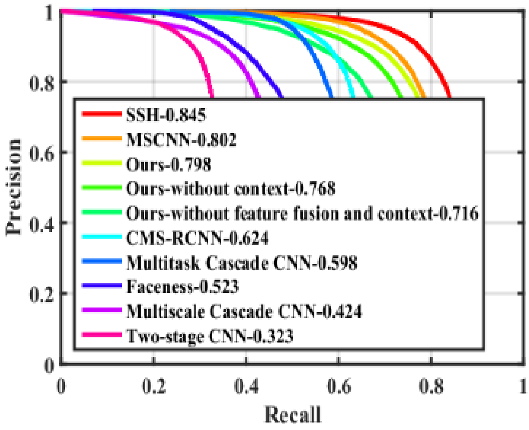
The precision-recall curves are shown in Figure 5.



(**a**)

(**b**)

Figure 5. *Cont.*

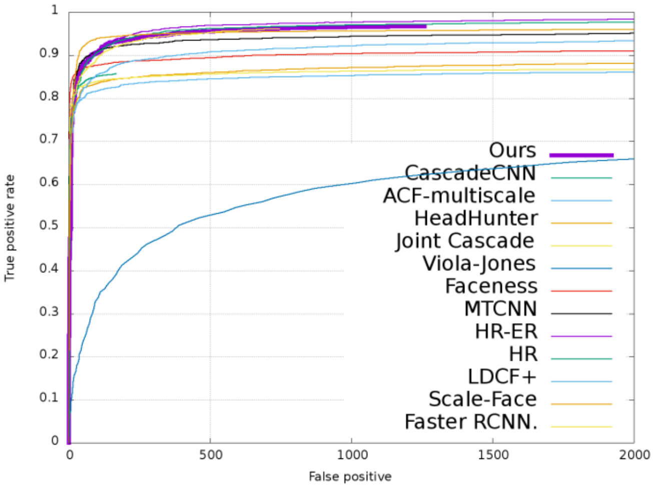


(**c**)

Figure 5. Precision-recall curves on WIDER FACE validation set. (**a**) Precision-recall curves on 'easy' subset; (**b**) Precision-recall curves on 'medium' subset; (**c**) Precision-recall curves on 'hard' subset.

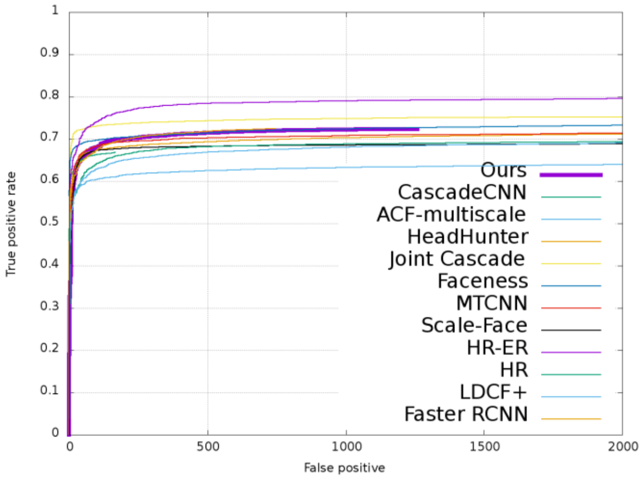
4.3. Evaluation Results on FDDB

The continuous ROC (Receiver operating characteristic, ROC) curves and discontinuous ROC curves evaluated on FDDB are shown in Figure 6. We obtain competitive performance on discontinuous ROC curves. The HR [5] is a recent ResNet-101 based method, and HR-ER [5] is a version of HR which additionally uses post-hoc regressor when evaluated on FDDB. Our ResNet-18 based lightweight method obtains nearly the same performance as HR-ER on discontinuous ROC curves, and surpasses HR on both discontinuous ROC curves and continuous ROC curves. However, our method's detection speed is only 9 ms but the HR-ER's is as high as 200 ms. Another recent VGG-16 based method DeepIR [33] performs better than ours on FDDB, and so we will further improve our method to catch up with DeepIR in the future. But it requires a complex training approach and uses many more anchors, and thus makes it much slower than ours.



(**a**)

Figure 6. *Cont.*



(**b**)

Figure 6. Receiver operating characteristic (ROC) curves on FDDB dataset. (**a**) Discontinuous ROC curves; (**b**) Continuous ROC curves.

4.4. Inference Time on Different Resolutions

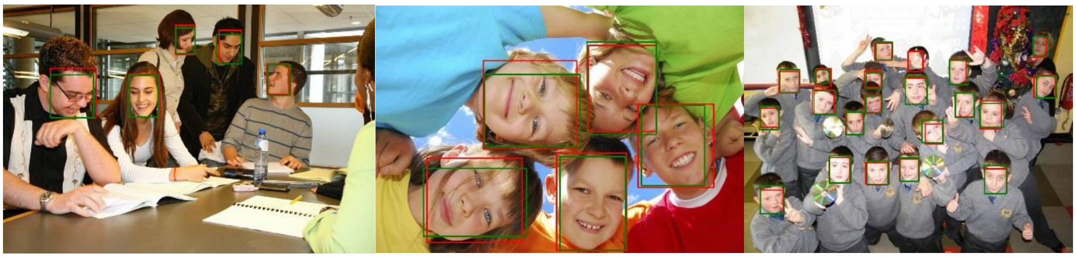
We further evaluate the inference speed of our method on different image resolutions. The results are shown in Table 2. Our method can obtain 303 fps on 288 × 288 images, 150 fps on 416 × 416 images and 40 fps on 800 × 1200 images. Our method is proved to be able to realize real time detection even on high resolution images.

Table 2. Inference time.

Image Size	Time	Fps
288 × 288	3.3 ms	303
416 × 416	5.7 ms	150
800 × 1200	25.4 ms	40
1200 × 1600	41.93 ms	24

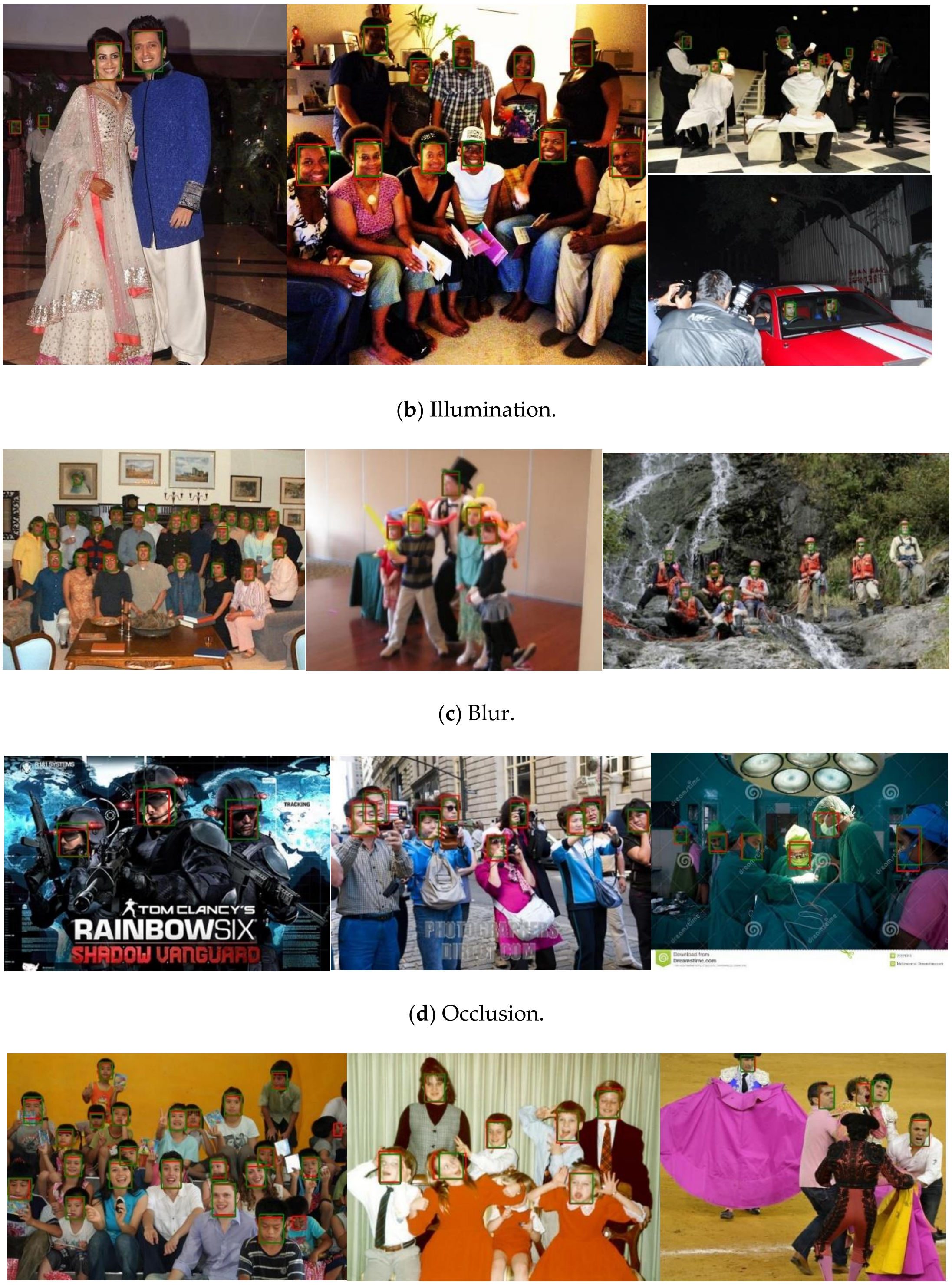
4.5. Qualitative Results

Qualitative results are shown in Figure 7, where we plot the ground truth bounding boxes in red and the predicted bounding boxes in green. To show the robustness of our method, we validate it on images under different conditions, i.e., pose, illumination, occlusion, racial factor and etc. As can be seen in Figure 7, our method has high overlap ratio between ground-truth and predicted bounding boxes, and it is robust under different challenging conditions.



(**a**) Pose changes.

Figure 7. *Cont.*



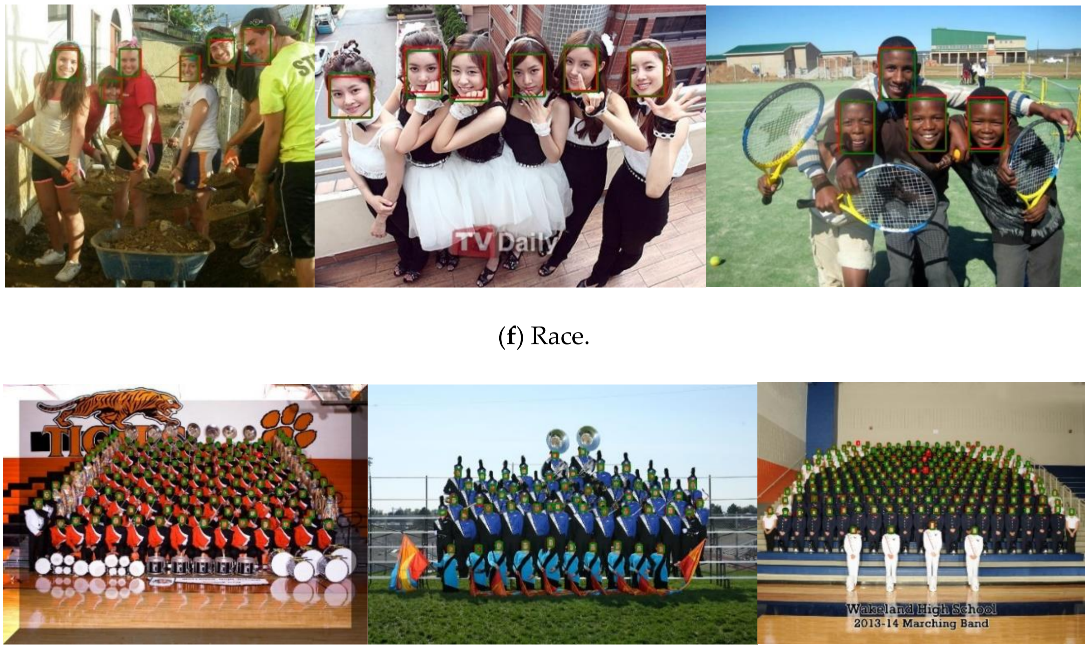
(**b**) Illumination.

(**c**) Blur.

(**d**) Occlusion.

(**e**) Expression.

Figure 7. *Cont.*



(**f**) Race.

(**g**) Small faces.

Figure 7. Qualitative results under various challenging conditions, i.e., illumination, pose changes, occlusion, race and etc. Ground truth bounding boxes are in red and predicted bounding boxes are in green. (Zoom in to see better).

5. Conclusions

In this paper, we propose an accurate, fast and lightweight face detection method based on a fast residual network. To detect multi-scale faces, we fuse feature maps of multiple convolutional layers within the network to generate multi-scale features for prediction. To further promote the detection accuracy, we add multi-context that includes local and global context to the fused multi-scale feature maps. Both of the multi-scale features and multi-context bring no computational burden to the backbone network. Experimental results show that our method can run at about 110 fps on VGA-resolution images. This speed is evaluated on a high-performance Titan X GPU, but the 110 fps implies that our method will still get a fast speed on a cheap and ordinary GPU. And it achieves high precision comparable to the state-of-the-art counterparts on WIDER FACE and FDDB dataset. In some cases, our lightweight method can even outperform the 'heavy' VGG-16 and ResNet-101 based methods. The lightweight characteristic and high precision make our method practical for deployment in real life applications. In the future, we will try to use some quantization and pruning approaches to compress our network, to further reducing the model size and accelerating the detection speed. But our method still has a gap comparing to best face detection methods in some complicated conditions, and the model size and computation still constrain our method deployed in embedded systems. In the future, we will try to improve the precision by designing a more powerful backbone network. And we will exploit quantization and pruning approaches to compress our network for further reducing model size and computation.

Author Contributions: Conceptualization, L.Z.; Data curation, L.Z.; Funding acquisition, X.Z.; Methodology, L.Z.; Project administration, X.Z.; Writing–original draft, L.Z.; Writing–review & editing, X.Z.

Funding: This work was supported by Shanghai Science and Technology Commission Research Project (Grant No. 16010500400), the Shanghai Innovation Action Plan Project (Grant No. 16511101200) and the National Key Research and Development Plan of China (Grant No. 2016YFB1000600 and 2016YFB1000601).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Chen, D.; Ren, S.; Wei, Y.; Cao, X.; Sun, J. Joint cascade face detection and alignment. In *European Conference on Computer Vision*; Springer: Cham, Switzerland, 2014; pp. 109–122.
2. Zhang, S.; Zhu, X.; Lei, Z.; Shi, H.; Wang, X.; Li, S.Z. S3fd: Single shot scale invariant face detector. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Venice, Italy, 22–29 October 2017; pp. 192–201.
3. Girshick, R. Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 13–16 December 2015; pp. 1440–1448.
4. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 24–27 June 2014; pp. 580–587.
5. Hu, P.; Ramanan, D. Finding tiny faces. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1522–1530.
6. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A.C. SSD: Single shot multibox detector. In *European Conference on Computer Vision*; Springer: Cham, Switzerland, 2016; pp. 21–37.
7. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
8. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. *arXiv* **2016**, arXiv:1512.03385.
9. Pang, Y.; Ye, L.; Li, X.; Pan, J. Incremental Learning with Saliency Map for Moving Object Detection. *IEEE Trans. Circuits Syst. Video Technol.* **2018**, *28*, 640–651. [CrossRef]
10. Chen, B.-H.; Shi, L.-F.; Ke, X. A Robust Moving Object Detection in Multi-Scenario Big Data for Video Surveillance. *IEEE Trans. Circuits Syst. Video Technol.* **2018**. [CrossRef]
11. Redmon, J.; Farhadi, A. Yolo9000: Better, faster, stronger. *arXiv* **2017**, arXiv:1612.08242.
12. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 8–10 June 2015; pp. 3431–3440.
13. Bell, S.; Zitnick, C.L.; Bala, K.; Girshick, R. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2874–2883.
14. Zagoruyko, S.; Lerer, A.; Lin, T.-Y.; Pinheiro, P.O.; Gross, S.; Chintala, S.; Dollar, P. A multipath network for object detection. *arXiv* **2016**, arXiv:1604.02135.
15. Zhu, C.; Zheng, Y.; Luu, K.; Savvides, M. CMS-RCNN: Contextual multi-scale region-based CNN for unconstrained face detection. *arXiv* **2017**, 57–79, arXiv:1606.05413.
16. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
17. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*; IEEE Computer Society: Washington, DC, USA, 2015; pp. 91–99.
18. Zhu, Q.; Yeh, M.C.; Cheng, K.T.; Avidan, S. Fast human detection using a cascade of histograms of oriented gradients. In Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, New York, NY, USA, 17–22 June 2006; pp. 1491–1498.
19. Viola, P.; Jones, M. Rapid object detection using a boosted cascade of simple features. In Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2001, Kauai, HI, USA, 8–14 December 2001; p. 1.
20. Li, H.; Lin, Z.; Shen, X.; Brandt, J.; Hua, G. A convolutional neural network cascade for face detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 5325–5334.
21. Yang, S.; Luo, P.; Loy, C.-C.; Tang, X. From facial parts responses to face detection: A deep learning approach. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 3676–3684.

22. Najibi, M.; Samangouei, P.; Chellappa, R.; Davis, L.S. SSH: Single stage headless face detector. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4875–4884.
23. Lin, T.-Y.; Dollar, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; p. 4.
24. Liu, W.; Rabinovich, A.; Berg, A.C. ParseNet: Looking wider to see better. *arXiv* **2015**, arXiv:1506.04579.
25. Deng, J.; Dong, W.; Socher, R.; Li, L.; Li, K.; Li, F.-F. ImageNet: A large-scale hierarchical image database. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
26. Pinheiro, P.O.; Lin, T.-Y.; Collobert, R.; Dollar, P. Learning to refine object segments. *arXiv* **2016**, arXiv:1603.08695.
27. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
28. Yang, S.; Luo, P.; Loy, C.-C.; Tang, X. Wider face: A face detection benchmark. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 5525–5533.
29. Jain, V.; Learned-Miller, E.G. *FDDB: A Benchmark for Face Detection in Unconstrained Settings*; UMass Amherst Technical Report; University of Massachusetts: Amherst, MA, USA, 2010.
30. Triantafyllidou, D.; Nousi, P.; Tefas, A. Fast deep convolutional face detection in the wild exploiting hard sample mining. *Big Data Res.* **2018**, *11*, 65–76. [CrossRef]
31. Cai, Z.; Fan, Q.; Feris, R.S.; Vasconcelos, N. A unified multi-scale deep convolutional neural network for fast object detection. In *European Conference on Computer Vision*; Springer: Berlin, Germany, 2016.
32. Zhang, K.; Zhang, Z.; Li, Z.; Qiao, Y. Joint face detection and alignment using multi-task cascaded convolutional networks. *IEEE Signal Process. Lett.* **2016**, *23*, 1499–1503. [CrossRef]
33. Sun, X.; Wu, P.; Hoi, S.C. Face detection using deep learning: An improved faster RCNN approach. *Neurocomputing* **2018**, *299*, 42–50. [CrossRef]