Robust Face Detection via Learning Small Faces on Hard Images

Zhishuai Zhang[1] Wei Shen[1] Siyuan Qiao[1] Yan Wang[1] Bo Wang[2] Alan Yuille[1]
Johns Hopkins University[1] Stanford University[2]
{zhshuai.zhang,shenwei1231,joe.siyuan.qiao,wyanny.9,wangbo.yunze,alan.l.yuille}@gmail.com

Abstract

Recent anchor-based deep face detectors have achieved promising performance, but they are still struggling to detect hard faces, such as small, blurred and partially occluded faces. One reason is that they treat all images and faces equally, and ignore the imbalance between easy images and hard images; however large amounts of training images only contain easy faces, which are less helpful to learn robust detectors for hard faces. In this paper, we propose that the robustness of a face detector against hard faces can be improved by learning small faces on hard images. Our intuitions are (1) hard images are the images which contain at least one hard face, thus they facilitate training robust face detectors; (2) most hard faces are small faces and other types of hard faces can be easily shrunk to small faces. To this end, we build an anchor-based deep face detector, which only outputs a single high-resolution feature map with small anchors, to specifically learn small faces and train it by a novel hard image mining strategy which automatically adjusts training weights on images according to their difficulties. Extensive experiments have been conducted on WIDER FACE, FDDB, Pascal Faces, and AFW datasets and our method achieves APs of 95.7, 94.9 and 89.7 on `easy`*,* `medium` *and* `hard` *WIDER FACE* `val` *dataset respectively, which verify the effectiveness of our methods, especially on detecting hard faces. Our detector is also lightweight and enjoys a fast inference speed. Code and model are available at https://github.com/bairdzhang/smallhardface.*

1. Introduction

Face detection is a fundamental and important computer vision problem, which is critical for many face-related tasks, such as face alignment [3, 36], tracking [9] and recognition [19, 24]. Stem from the recent success of deep neural networks, massive CNN-based face detection approaches [7, 17, 29, 42, 45, 46] have been proposed and achieved state-of-the-art performance. However, face detection remains a challenging task due to occlusion, illumination, makeup, as well as pose and scale variance, as shown in the benchmark dataset WIDER FACE [40].

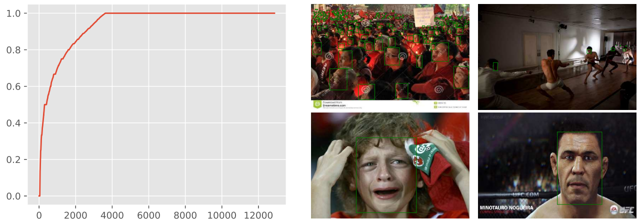


Figure 1: Left: AP of each training image computed based on official SSH model, the x-axis is index of the training image, the y-axis is the AP for the corresponding image. Upper right: hard training images. Lower right: easy training images.

Current state-of-the-art CNN-based face detectors attempt to address these challenges by employing more powerful backbone models [1], exploiting feature pyramid-style architectures to combine features from multiple detection feature maps [29], designing denser anchors [46] and utilizing larger contextual information [29]. These methods and techniques have been shown to be successful to build a robust face detector, and improve the performance towards human-level for most images.

In spite of their success for most images, an evident performance gap still exists especially for those hard images which contain small, blurred and partially occluded faces. We realize that these hard images have become the main barriers for face detectors to achieve human-level performance. In Figure 1, we show that, even on the `train` set of WIDER FACE, the official SSH model [1] still fails on some of the images with extremely hard faces. We show two such hard training images in the upper right corner in Figure 1.

On the contrary, most training images with easy faces can be almost perfectly detected (see the illustration in the lower right corner of Figure 1). As shown in the left part of Figure 1, over two-thirds of the training images already obtained perfect detection accuracy, and we believe that those easy images are less useful towards training a robust face detector. To address this issue, in this paper, we propose a

[1] https://github.com/mahyarnajibi/SSH

robust face detector by putting more training focus on those hard images.

This problem is mostly related to anchor-level hard example mining discussed in OHEM [26]. OHEM aims at mining hard anchors/proposals in each image. However, when the majority of the training set are easy images with no hard anchors/proposals (see Figure 1), OHEM will exhibit less effectiveness since less useful information can be exploited towards a more robust model. To this end, we propose to mine hard positive examples at image level (*i.e.* hard image mining, HIM) in conjunction with anchor level OHEM. More specifically, we propose to dynamically assign difficulty scores to training images during the learning process, which can determine whether an image is already well-detected or still useful for further training. This allows us to fully utilize the images which were not perfectly detected to better facilitate the subsequent learning process. We show that this strategy can make our detector more robust towards hard faces, without involving more complex network architecture and computation overhead. The main difference between our HIM and OHEM is that ours takes an image as a unit for mining while OHEM takes an anchor/proposal as a unit for mining. Our HIM is complementary to OHEM and our face detector uses both OHEM and HIM since they aim to solve different issues.

Apart from mining the hard images, we also propose to improve the detection quality by exclusively exploiting small faces. Small faces are typically hard and have attracted extensive research attention [1, 7, 46]. Existing methods aim at building a scale-invariant face detector to learn and infer on both small and big faces, with multiple levels of detection features and anchors of different sizes. Compared with these methods, our detector is more efficient since it is specially designed to aggressively leverage the small faces during training. More specifically, large faces are automatically ignored during training due to the design of small anchors and single high-resolution feature map, so that the model can fully focus on the small hard faces. Experimental results also demonstrate the effectiveness of our design.

To conclude, in this paper, we propose a novel face detector with the following contributions:

- We propose a hard image mining strategy, to improve the robustness of our detector to those extremely hard faces. This is done without any extra modules, parameters or computation overhead added on the existing detector.

- We design a single shot detector with only one detection feature map, which focuses on small faces with a specific range of sizes. This allows our model to be simple and focus on difficult small faces without struggling with scale variance.

- Our face detector achieves state-of-the-art level performance on all popular face detection benchmarks, including WIDER FACE, FDDB, Pascal Faces, and AFW. We achieve 95.7, 94.9 and 89.7 on `easy`, `medium` and `hard` WIDER FACE `val` dataset. Our method also achieves APs of 99.00 and 99.60 on Pascal Faces and AFW respectively, as well as a TPR of 98.7 on FDDB.

The remainder of this paper is organized as follows. In Section 2, we discuss some studies which are related to our paper. In Section 3, we dive into details of our proposed method, and we discuss experimental results and ablation experiments in Section 4. Finally, conclusions are drawn in Section 5.

2. Related work

Face detection has received extensive research attention [11, 16, 31]. With the emergence of modern CNN [6, 10, 27] and object detector [4, 14, 21, 22], there are many face detectors proposed to achieve promising performances [17, 29, 33, 34, 45], by adopting general object detection framework into face detection domain. In this section, we briefly review hard example mining, face detection architecture, and anchor design & matching.

2.1. Hard example mining

Hard example mining is an important strategy to improve model quality, and has been studied extensively in image classification [15] and general object detection [13, 26]. The main idea is to find some hard positive and hard negative examples at each step, and put more effort into training on those hard examples [23, 30]. Recently, with modern detection frameworks proposed to boost the performance, OHEM [26] and Focal Loss [13] have been proposed to select hard examples. OHEM computed the gradients of the networks by selecting the proposals with highest losses in every minibatch; while Focal Loss aimed at naturally putting more focus on hard and misclassified examples by adding a factor to the standard cross entropy criterion. However, these algorithms mainly focused on anchor-level or proposal-level mining, and they will exhibit less effectiveness on easy images where there are no hard anchors/proposals. In face detection, most training images are easy as shown in Figure 1, while some extremely hard images drag the final performance down. To this end, we propose to exploit hard example mining on image level, *i.e.* hard image mining, to improve the quality of face detector on extremely hard faces. Our hard image mining can be used in conjunction with OHEM, to put more training efforts on hard images and make OHEM more effective.

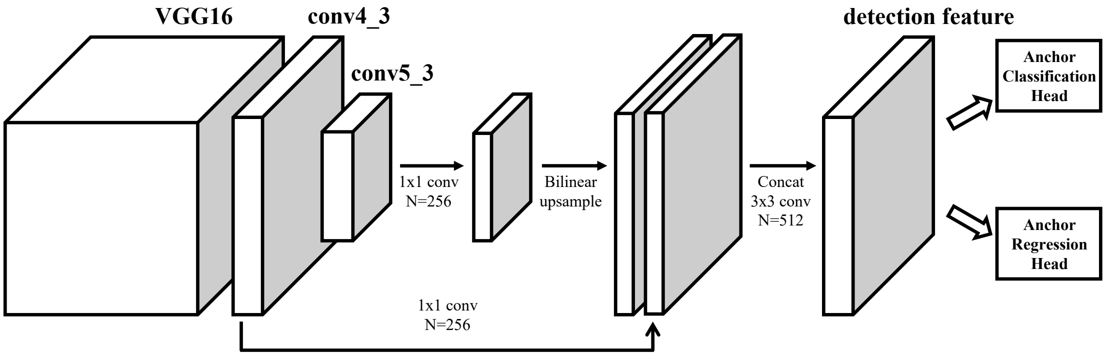


Figure 2: The framework of our face detector. We take VGG16 as our backbone CNN, and we fuse two layers (conv4_3 and conv5_3) after dimension reduction and bilinear upsampling, to generate the final detection feature map. Based on that, we add a detection head for classification and bounding-box regression.

2.2. Face detection architecture

Recent state-of-the-art face detectors are generally built based on Faster-RCNN [22], R-FCN [4] or SSD [14]. SSH [17] exploited the RPN (Region Proposal Network) from Faster-RCNN to detect faces, by building three detection feature maps and designing six anchors with different sizes attached to the detection feature maps. S^{3}FD [45] and PyramidBox [29], on the other hand, adopted SSD as their detection architecture with six different detection feature maps. Different from S^{3}FD, PyramidBox exploited a feature pyramid-style structure to combine features from different detection feature maps. Our proposed method, on the other hand, only builds single level detection feature map, based on VGG16, for classification and bounding-box regression, which is both simple and effective.

2.3. Anchor design and matching

Usually, anchors are designed to have different sizes to detect objects with different scales, in order to build a scale-invariant detector. SSD as well as its follow-up detectors S^{3}FD and PyramidBox, had six sets of anchors with different sizes, ranging from (16 × 16) to (512 × 512), and their network architectures had six levels of detection feature maps, with resolutions ranging from $\frac{1}{4}$ to $\frac{1}{128}$, respectively. Similarly, SSH had the same anchor setting, and those anchors were attached to three levels of detection feature maps with resolutions ranging from $\frac{1}{8}$ to $\frac{1}{32}$. The difference between SSH and S^{3}DF is that in SSH, anchors with two neighboring sizes share the same detection feature map, while in S^{3}DF, anchors with different sizes are attached to different detection feature maps.

SNIP [28] discussed an alternative approach to handle scales. It showed that CNNs are not robust to changes in scale, so training and testing on the same scales of an image pyramid can be a more optimal strategy. In our paper, we exploit this idea by limiting the anchor sizes to be (16 × 16), (32 × 32) and (64 × 64). Then those faces with either too small or too big sizes will not be matched to any of the anchors, thus will be ignored during the training and testing. By removing those large anchors with sizes larger than (64 × 64), our network focuses more on small faces which are potentially more difficult. To process large faces, we use multi-scale training and testing to resize them to match our small anchors.

3. Proposed method

In this section, we introduce our proposed method for effective face detection. We first discuss the architecture of our detector in Subsection 3.1, then we elaborate our hard image mining strategy in Subsection 3.2, as well as some other useful training techniques in Subsection 3.3.

3.1. Single-level small face detection framework

The framework of our face detector is illustrated in Figure 2. We use VGG16 network as our backbone CNN, and combine conv4_3 and conv5_3 features, to build the detection feature map with both low-level and high-level semantic information. Similar to SSH [17], we apply 1×1 convolution layers after conv4_3 and conv5_3 to reduce dimension, and then apply a 3×3 convolution layer on the concatenation of these two dimension reduced features. The output feature of the 3×3 convolution layer is the final detection feature map, which will be fed into the detection head for classification and bounding-box regression.

The detection feature map has a resolution of $\frac{1}{8}$ of the original image (of size $H \times W$). We attach three anchors at each point in the grid as default face detection boxes. Then we do classification and bounding-box regression on those $3 \times \frac{H}{8} \times \frac{W}{8}$ anchors. Unlike many other face detectors which build multiple feature maps to detect faces with a variant

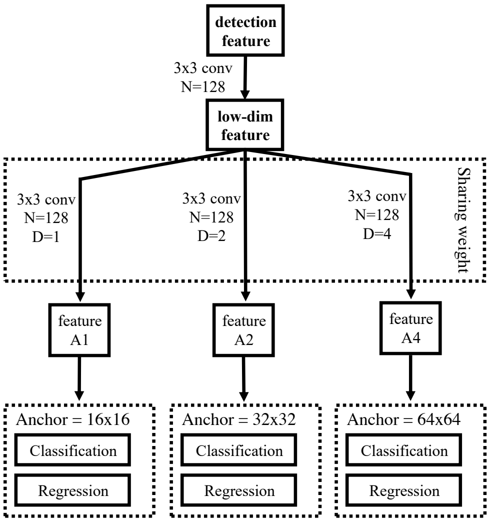


Figure 3: The framework of our dilated detection head for classification and regression. Based on the detection feature map from the backbone CNN, we first perform dimension reduction to reduce the number of channels from 512 to 128. Then we put three convolution layers with the shared weight and different dilation rates, to generate final detection and classification features.

range of scales, inspired by SNIP [28], faces are trained and inferred with roughly the same scales. We only have one detection feature map, with three sets of anchors attached to it. The anchors have sizes of (16×16), (32×32) and (64×64), and the aspect ratio is set to be 1. By making this configuration, our network only trains and infers on faces with small and medium sizes; and we propose to handle large faces by shrinking the images in the test phase. We argue that there is no speed or accuracy degradation for large faces, since inferring on a tiny image (with the short side containing 100 or 300 pixels) is very fast, and the shrunk large faces will still have enough information to be recognized.

To handle the difference of anchor sizes attached to the same detection feature map, we propose a detection head which uses different dilation rates for anchors with different sizes, as shown in Figure 3. The intuition is that in order to detect faces with different sizes, different effective receptive field sizes are required. This naturally requires the backbone feature map to be invariant to scales. To this end, we adopt different dilation rates for anchors with different sizes. For anchors with size (16×16), (32×32) and (64×64), we use a convolution with a kernel size of 3 and dilation rate of 1, 2 and 4 to gather context features at different scales. These three convolution layers share weights to reduce the model size. With this design, the input of the 3×3 convolution, will be aligned to the same location of faces, regardless of the size of faces and anchors. Subsection 4.3 will show the effectiveness of this multi-dilation design.

3.2. Hard image mining

Different from OHEM, which selects proposals or anchors with the highest losses, we propose a novel hard image mining strategy at the image level. The intuition is that most images in the dataset are very easy, and we can achieve a very high AP even on the `hard` subset of the WIDER FACE `val` dataset with our baseline model, while there are still some extremely challenging images with occlusion, illumination, makeup and pose/scale variance. We believe not all training images should be treated equally, and well-recognized images will not help towards training a more robust face detector. To put more attention on training hard images instead of easy ones, we use a subset $\mathcal{D}'$ of all training images $\mathcal{D}$, to contain hard ones for training. At the beginning of each epoch, we build $\mathcal{D}'$ based on the difficulty scores obtained in the previous epoch.

We initially use all training images to train our model (*i.e.* $\mathcal{D}' = \mathcal{D}$). This is due to the fact that our initial ImageNet pre-trained model will only give random guess towards face detection. In this case, there is no easy image and, every image is considered as hard image and fed to the network for training at the first epoch. During the training procedure, we dynamically assign different difficulty scores to training images, which is defined by the metric Worst Positive Anchor Score (WPAS):

$$\text{WPAS}(I;\Theta) = \min_{a \in \mathcal{A}(I)^+} \frac{\exp(l(I;\Theta)_{a,1})}{\exp(l(I;\Theta)_{a,1}) + \exp(l(I;\Theta)_{a,0})}$$

where $\mathcal{A}(I)^+$ is the set of positive anchors for image I, with IoU over 0.5 against ground-truth boxes, l is the classification logit and $l(I;\Theta)_{a,1}$, $l(I;\Theta)_{a,0}$ are the logits of anchor a for the image I to be foreground face and background. All images are initially marked as hard, and any image with WPAS greater than a threshold th will be regarded as easy, since all positive anchors have been correctly recognized.

At the beginning of each epoch, we first randomly shuffle the training dataset to generate the complete training list $\mathcal{D} = [I_{i_1}, I_{i_2}, \cdots, I_{i_n}]$ for the following epoch of training, where $i_1, \cdots, i_n$ is a random permutation of $1, \cdots, n$. Then given an image marked as easy, we remove it from $\mathcal{D}$ with a probability of p. The remaining training list $\mathcal{D}' = [I_{i_{j_i}}, I_{i_{j_2}}, \cdots, I_{i_{j_k}}]$ (a sublist of $\mathcal{D}$), which focuses more on hard images, will be used for training at this epoch. Note that for multi-GPU training, each GPU will maintain its training list $\mathcal{D}'$ independently. In our experiments, we set the probability p to 0.7, and the threshold th to 0.85. We test with different p and th and find that the model performs consistently well when th falls between 0.3 and 0.85 and p falls between 0.5 and 0.7.

3.3. Training strategy

Multi-scale training and anchor matching

Since we only have anchors covering a limited range of face scales, we train our model by varying the sizes of training images. During the training phase, we resize the training images so that the short side of the image contains s pixels, where s is randomly selected from $\{400, 800, 1200\}$. We also set an upper bound of 2000 pixels to the long side of the image considering the GPU memory limitation.

For each anchor, we assign a label $\{+1, 0, -1\}$ based on how well it matches with any ground-truth face bounding box. If an anchor has an IoU (Intersection over Union) over 0.5 against a ground-truth face bounding box, we assign $+1$ to that anchor. On the other hand, if the IoU against any ground-truth face bounding box is lower than 0.3, we assign 0 to that anchor. All other anchors will be given -1 as the label, and thus will be ignored in the classification loss. By doing so, we only train on faces with designated scales. Those faces with no anchor matching will be simply ignored, since we do not assign the anchor with largest IoU to it (thus assign the corresponding anchor label $+1$) as Faster-RCNN does. This anchor matching strategy will ignore the large faces, and our model can put more capacity on learning different face patterns on hard small faces instead of memorizing the change in scales.

For the regression loss, all anchors with IoU greater than 0.3 against ground-truth faces will be taken into account and contribute to the smooth $\ell 1$ loss. We use a smaller threshold (*i.e.* 0.3) because (1) this will allow imperfectly matched anchors to be able to localize the face, which may be useful during the testing and (2) the regression task has less supervision since unlike classification, there are no negative anchors for computing loss and the positive anchors are usually sparse.

Anchor-level hard example mining

OHEM has been proven to be useful for object detection and face detection in [14, 17, 26]. During our training, in parallel with our newly proposed hard image mining, we also use the traditional hard anchor mining method to focus more on the hard and misclassified anchors. Given a training image with size $H \times W$, there are $3 \times \frac{H}{8} \times \frac{W}{8}$ anchors at the detection head, and we only select 256 of them to be involved in computing the classification loss. For all positive anchors with IoU greater than 0.5 against ground-truth boxes, we select the top 64 of them with lowest confidences to be recognized as faces. After selecting positive anchors, $(256 - \#pos\_anchor)$ negative anchors with highest face confidences are selected to compute the classification loss as hard negative anchors. Note that we only perform OHEM for classification loss, and we keep all anchors with IoU greater than 0.3 for computing regression loss, without selecting a subset based on either classification loss or bounding-box regression loss.

Data augmentation

Data augmentation is extremely useful to make the model robust to light, scale changes and small shifts [14, 29]. In our proposed method, we exploit cropping and photometric distortion as data augmentation. Given a training image after resizing, we crop a patch of it with a probability of 0.5. The patch has a height of H' and a width of W' which are independently drawn from $\mathcal{U}(0.6H, H)$ and $\mathcal{U}(0.6W, W)$, where $\mathcal{U}$ is the uniform distribution and H, W are the height and width of the resized training image. All ground-truth boxes whose centers are located inside the patch are kept. After the random cropping, we apply photometric distortion following SSD by randomly modifying the brightness, contrast, saturation and hue of the cropped images.

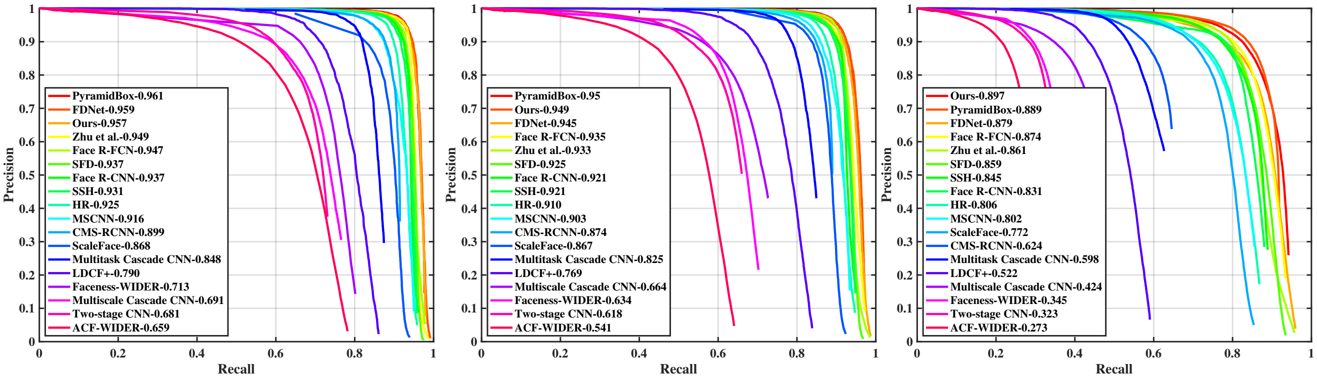
4. Experiments

To verify the effectiveness of our model and proposed method, we conduct extensive experiments on popular face detection benchmarks, including WIDER FACE [40], FDDB [8], Pascal Faces [37] and AFW [48]. It is worth noting that the training is only performed on the `train` set of WIDER FACE, and we use the same model for evaluation on all these datasets without further fine-tuning.

4.1. Experiment settings

We train our model on the `train` set of WIDER FACE, which has 12880 images with 159k faces annotated. We flip all images horizontally, to double the size of our training dataset to 25760. For each training image, we first randomly resize it, and then we use the cropping and photometric distortion data augmentation methods discussed in Section 3.3 to pre-process the resized image. We use an ImageNet pretrained VGG16 [10] model to initialize our network backbone, and our newly introduced layers are randomly initialized with Gaussian initialization. We train the model with the itersize to be 2, for 46k iterations, with a learning rate of 0.004, and then for another 14k iterations with a smaller learning rate of 0.0004. For training, we use 4 GPUs to simultaneously to compute the gradient and update the weight by synchronized SGD with Momentum [20]. The first two blocks of VGG16 are frozen during the training, and the rest layers of VGG16 are set to have a double learning rate.

Since our model is designed and trained on only small faces, we use a multiscale image pyramid for testing to deal with faces larger than our anchors. Specifically, we resize the testing image so that the short side contains 100, 300, 600, 1000 and 1400 pixels for evaluation on WIDER FACE



easy subset

medium subset

hard subset

Figure 4: Precision-recall curve on WIDER FACE val dataset.

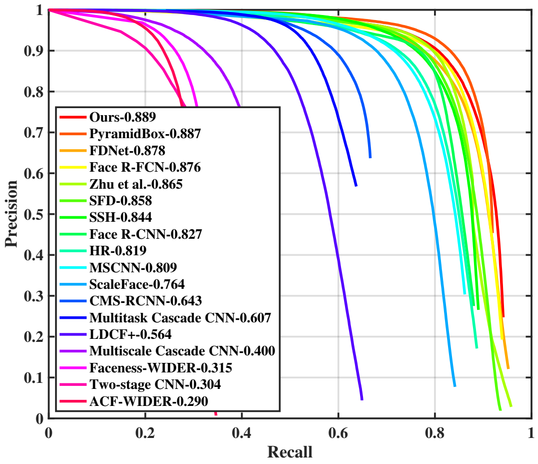


Figure 5: Precision-recall curve on the hard subset of WIDER FACE test dataset.

dataset. We also follow the testing strategies used in PyramidBox [29][2] such as horizontal flip and bounding-box voting [5].

4.2. Experiment results

WIDER FACE dataset includes 3226 images and 39708 faces labelled in the val dataset, with three subsets – easy, medium and hard. In Figure 4, we show the precision-recall (PR) curve and average precision (AP) for our model compared with many other state-of-the-arts [1, 2, 17, 18, 29, 32, 33, 34, 35, 38, 39, 40, 41, 43, 44, 45, 46, 47] on these three subsets. As we can see, our method achieves the state-of-the-art level accuracy on the hard subset. Since the hard set is a superset of small and medium, which contains all faces taller than 10 pixels, the performance on hard set can represent the performance on the full testing dataset more accurately. Our performance on the medium subset is comparable to the most recent state-of-the-art and the performance on the easy subset is a bit worse since our method focuses on learning hard faces, and the architecture of our model is simpler compared with other state-of-the-arts.

There is also a WIDER FACE test dataset with no annotations provided publicly. It contains 16097 images, and is evaluated by WIDER FACE author team. We report the performance of our method at Figure 5 for the hard subset.

FDDB dataset includes 5171 faces on a set of 2845 images, and we use our model trained on WIDER FACE train set to infer on the FDDB dataset. We use the raw bounding-box result without fitting it into ellipse to compute ROC. We show the discontinuous ROC curve at Figure 6a compared with [12, 16, 25, 29, 31, 37, 43, 45, 48], and our method achieves the state-of-the-art performance with a TPR of 98.7% given 1000 false positives.

Pascal Faces dataset includes 1335 labeled faces on a set of 851 images extracted for the Pascal VOC dataset. We show the PR curve at Figure 6b compared with [16, 45], and our method achieves a new state-of-the-art performance of AP=99.0.

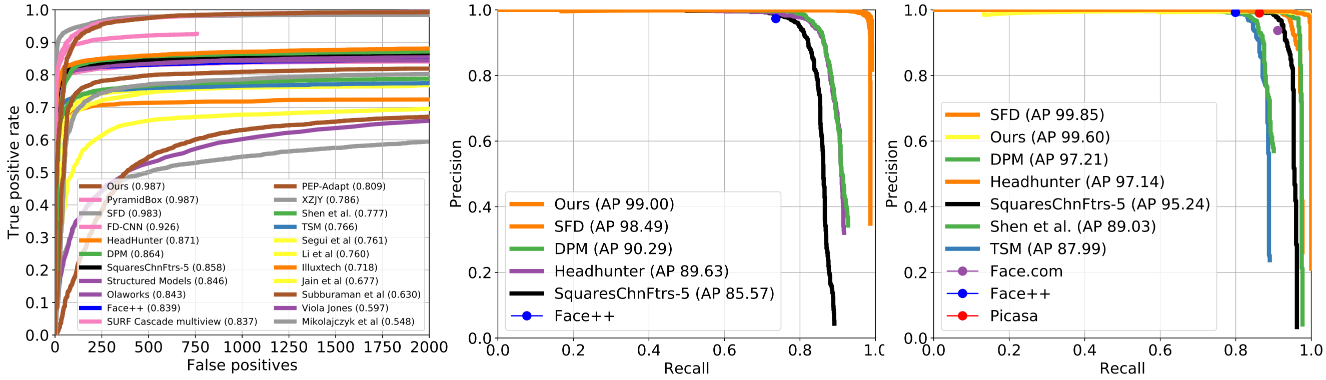
AFW dataset includes 473 faces labelled in a set of 205 images. As shown in Figure 6c compared with [16, 25, 45, 48], our method achieves state-of-the-art and almost perfect performance, with an AP of 99.60.

4.3. Ablation study and diagnosis

Ablation experiments

In order to verify the performance of our single level face detector, as well as the effectiveness of our proposed hard image mining, the dilated-head classification and regression structure, we conduct various ablation experiments on the WIDER FACE val dataset. All results are summarized in Table 1. From Table 1, we can see that our single level baseline model can already achieve performance compara-

[2] https://github.com/PaddlePaddle/models/blob/develop/fluid/PaddleCV/face_detection/widerface_eval.py



(a) FDDB

(b) Pascal Faces

(c) AFW

Figure 6: Performance compared with state-of-the-arts on other face datasets.

Method	easy	medium	hard
Baseline-Three	95.0	93.8	88.5
+ HIM	95.5	94.5	89.0
Baseline-Single	95.1	94.2	89.1
+ HIM	95.4	94.8	89.6
+ DH	95.4	94.5	89.3
+ DH + HIM	**95.7**	**94.9**	**89.7**

Table 1: Ablation experiments. Baseline-Three is a face detector similar to SSH with three detection feature maps. Baseline-Single is our proposed detector with single detection feature map shown in Figure 2. HIM and DH represents hard image mining (Subsection 3.2) and dilated head architecture (Figure 3). The same testing strategy is used.

ble to the current state-of-the-art face detector, especially on the hard subset. Our model with single detection feature map performs better than the one with three detection feature maps, despite its shallower structure, fewer parameters and anchors. This confirms the effectiveness of our simple face detector with single detection feature map focusing on small faces.

We also separately verify our newly proposed hard image mining (HIM) and dilated head architecture (DH) described in Subsection 3.2 and Figure 3 respectively. HIM can improve the performance on hard subset significantly without involving more complex network architecture nor computation overhead, and this benefit holds for both our three-map baseline and single-map baseline. DH itself can also boost the performance, which shows the effectiveness of designing larger convolution for larger anchors. Combining HIM and DH together can achieve the best performance.

Diagnosis of OHEM as hard face mining

In this subsection we investigate the possibility of using online hard example mining (OHEM) [26] to make the network focus more on hard faces, by selecting hard proposals/anchors in each image separately. Instead of selecting the top 64 hardest positive anchors as we did in Subsection 3.3, we mimic the settings used in HIM (see 3.2), and disable a positive anchor with a probability of p if the score of that positive anchor is greater than th, where p is 0.7 and th is 0.85. All other settings are kept the same and we do not use the dilated head architecture (DH) here. The results are shown in Table 2, and the performance of using OHEM to mine hard faces (listed as OHEM_HARD_FACE in the table) is inferior compared with our HIM. It confirms our intuition that most images are easy without hard faces, and mining hard faces on those easy images is not helpful to train a more robust detector, so it is important to select and focus on hard images.

Method	easy	medium	hard
Baseline-Single	95.1	94.2	89.1
+ HIM	95.4	94.8	89.6
+ OHEM_HARD_FACE	95.0	94.2	89.1

Table 2: Diagnose of using OHEM to mine hard faces. All entries are based on our Baseline-SingleLevel configuration without DH.

Diagnosis of multi-scale testing

Our face detector with one detection feature map is designed for small face detection, and our anchors are only capable of capturing faces with sizes ranging from (16×16) to (64×64). As a result, it is critical to adopt multi-scale testing to deal with large faces. Different from SSH, S^{3}FD and PyramidBox, our testing pyramid includes some extremely small scales (*i.e.* short side contains only 100 or 300 pixels). In Table 3, we show the effectiveness of these extremely small scales to deal with easy and large images.

Testing Scales (x100)	`easy`	`medium`	`hard`
[6, 10, 14]	78.2	85.7	86.1
[3, 6, 10, 14]	91.3	92.6	88.8
[1, 3, 6, 10, 14]	95.7	94.9	89.7

Table 3: Diagnosis of multi-scale testing. All results are evaluated with the same model with HIM and DH.

Our full evaluation resizes the image so that the short side contains 100, 300, 600, 1000 and 1400 pixels respectively, to build an image pyramid. We diagnose the impact of the extra small scales (*i.e.* 100 and 300) by removing them from the image pyramid.

As shown in Table 3, the extra small scales are crucial to detect easy faces. Without resizing the short side to contain 100 and 300 pixels, the performance on `easy` subset is only 78.2, which is even lower than the performance on `medium` and `hard` which contain much harder faces. We will show in the next subsection that these extra small scales (100 and 300) lead to negligible computation overhead, due to the lower resolution.

Diagnosis of accuracy/speed trade-off

We evaluate the speed of our method as well as some other popular face detectors in Table 4. For fair comparison, we run all methods on the same machine, with one Titan X Maxwell GPU, and Intel Core i7-4770K 3.50GHz. All methods except for PyramidBox are based on Caffe1 implementation, which is compiled with CUDA 9.0 and CUDNN 7. For PyramidBox, we follow the official fluid code and the default configurations[3]. We use the officially built PaddlePaddle with CUDA 9.0 and CUDNN 7[4].

For SSH, S^{3}FD and Pyramid, we use the official inference code and configurations. For SSH, we use multi-scale testing with the short side containing 500, 800, 1200 and 1600 pixels, and for S^{3}FD, we execute the official evaluation code with both multi-scale testing and horizontal flip. PyramidBox takes a similar testing configuration as S^{3}FD. As shown in Table 4, our detector can outperform SSH, S^{3}FD and PyramidBox significantly with a smaller inference time. Based on that, using horizontal flip can further improve the performance slightly. In terms of GPU memory usage, our method uses only a half of what PyramidBox occupies, while achieving better performance.

Ours* in Table 4 indicates our method without extra small scales in inference, *i.e.*, evaluated with scales [600, 1000, 1400]. It is only 6.5% faster than evaluation with [100, 300, 600, 1000, 1400] (1.59 compared with 1.70). This proves that although our face detector is only trained on small faces, it can perform well on large faces, by simply shrinking the testing image with negligible computation overhead.

Method	MS	HF	Time	G-Mem	AP-h
SSH	Yes	No	1.00	6.1	84.5
S^{3}FD	Yes	Yes	1.34	6.2	85.2
PyramidBox	Yes	Yes	2.24	11.9	88.9
Ours*	Yes*	Yes	1.59	**5.3**	86.1
Ours	Yes	No	**0.84**	**5.3**	89.3
Ours	Yes	Yes	1.70	**5.3**	**89.7**

Table 4: Diagnosis of inference speed. MS and HF indicate multi-scale testing and horizontal flip; Time is the inference time (in second) for a single image; G-Mem is the GPU memory usage in gigabyte; AP-h is the average precision on the `hard` subset of WIDER FACE `val` set. Ours* indicates our detector without extra small scales. All entries are evaluated with a single nVIDIA Titan X Maxwell.

5. Conclusion

To conclude, we propose a novel face detector to focus on learning small faces on hard images, which achieves superior performance on all popular face detection benchmarks. We identify that there are many easy training images with no hard faces which are less useful to build a robust detector, and traditional online hard example mining (OHEM) is unable to handle this imbalance between images. Based on this, we propose a hard image mining strategy by dynamically assigning difficulty scores to training images, and re-sampling subsets with more hard images and fewer easy images for training before each epoch. We also design a single shot face detector with only one detection feature map, to train and test on small faces. With these designs, our model can put more attention on learning small hard faces instead of memorizing change of scales. Extensive experiments and ablations have been carried out to show the effectiveness of our method, and our face detector achieves the state-of-the-art performance on all popular face detection benchmarks, including WIDER FACE, FDDB, Pascal Faces and AFW. Our face detector also enjoys a fast multi-scale inference speed and less GPU memory usage. Our proposed method is flexible and can be applied to other backbones and tasks, which we remain as future work.

Acknowledgements

This research was supported by the Office of Naval Research with grant N00014-15-1-2356 and N00014-16-1-3168, as well as the National Natural Science Foundation of China No. 61672336.

[3] https://github.com/PaddlePaddle/models/blob/develop/fluid/PaddleCV/face_detection/widerface_eval.py

[4] `pip install paddlepaddle-gpu`

References

[1] Y. Bai, Y. Zhang, M. Ding, and B. Ghanem. Finding tiny faces in the wild with generative adversarial network. In *CVPR*, pages 21–30, 2018. 1, 2, 6

[2] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos. A unified multi-scale deep convolutional neural network for fast object detection. In *ECCV*, pages 354–370, 2016. 6

[3] X. Cao, Y. Wei, F. Wen, and J. Sun. Face alignment by explicit shape regression. *IJCV*, pages 177–190, 2014. 1

[4] J. Dai, Y. Li, K. He, and J. Sun. R-fcn: Object detection via region-based fully convolutional networks. In *NIPS*, pages 379–387, 2016. 2, 3

[5] S. Gidaris and N. Komodakis. Object detection via a multi-region and semantic segmentation-aware cnn model. In *ICCV*, pages 1134–1142, 2015. 6

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 2

[7] P. Hu and D. Ramanan. Finding tiny faces. In *CVPR*, pages 1522–1530, 2017. 1, 2

[8] V. Jain and E. Learned-Miller. Fddb: A benchmark for face detection in unconstrained settings. Technical report, 2010. 5

[9] M. Kim, S. Kumar, V. Pavlovic, and H. Rowley. Face tracking and recognition with visual constraints in real-world videos. In *CVPR*, pages 1–8, 2008. 1

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012. 2, 5

[11] H. Li, Z. Lin, J. Brandt, X. Shen, and G. Hua. Efficient boosted exemplar-based face detection. In *CVPR*, pages 1843–1850, 2014. 2

[12] J. Li and Y. Zhang. Learning surf cascade for fast and accurate object detection. In *CVPR*, pages 3468–3475, 2013. 6

[13] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *ICCV*, pages 2999–3007, 2017. 2

[14] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *ECCV*, pages 21–37, 2016. 2, 3, 5

[15] I. Loshchilov and F. Hutter. Online batch selection for faster training of neural networks. *arXiv preprint arXiv:1511.06343*, 2015. 2

[16] M. Mathias, R. Benenson, M. Pedersoli, and L. Van Gool. Face detection without bells and whistles. In *ECCV*, pages 720–735, 2014. 2, 6

[17] M. Najibi, P. Samangouei, R. Chellappa, and L. Davis. SSH: Single stage headless face detector. In *ICCV*, pages 4885–4894, 2017. 1, 2, 3, 5, 6

[18] 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. 6

[19] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *BMVC*, pages 41.1–41.12, 2015. 1

[20] N. Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, pages 145–151, 1999. 5

[21] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, pages 779–788, 2016. 2

[22] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99, 2015. 2, 3

[23] H. A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *PAMI*, pages 23–38, 1998. 2

[24] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, pages 815–823, 2015. 1

[25] X. Shen, Z. Lin, J. Brandt, and Y. Wu. Detecting and aligning faces by image retrieval. In *CVPR*, pages 3460–3467, 2013. 6

[26] A. Shrivastava, A. Gupta, and R. Girshick. Training region-based object detectors with online hard example mining. In *CVPR*, pages 761–769, 2016. 2, 5, 7

[27] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2

[28] B. Singh and L. S. Davis. An analysis of scale invariance in object detection–snip. In *CVPR*, pages 3578–3587, 2018. 3, 4

[29] X. Tang, D. K. Du, Z. He, and J. Liu. Pyramidbox: A context-assisted single shot face detector. In *ECCV*, pages 812–828, 2018. 1, 2, 3, 5, 6

[30] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, pages 511–518, 2001. 2

[31] P. Viola and M. J. Jones. Robust real-time face detection. *IJCV*, pages 137–154, 2004. 2, 6

[32] H. Wang, Z. Li, X. Ji, and Y. Wang. Face r-cnn. *arXiv preprint arXiv:1706.01061*, 2017. 6

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

[34] Y. Wang, X. Ji, Z. Zhou, H. Wang, and Z. Li. Detecting faces using region-based fully convolutional networks. *arXiv preprint arXiv:1709.05256*, 2017. 2, 6

[35] Y. Wang, X. Ji, Z. Zhou, H. Wang, and Z. Li. Detecting faces using region-based fully convolutional networks. *arXiv preprint arXiv:1709.05256*, 2017. 6

[36] X. Xiong and F. De la Torre. Supervised descent method and its applications to face alignment. In *CVPR*, pages 532–539, 2013. 1

[37] J. Yan, X. Zhang, Z. Lei, and S. Z. Li. Face detection by structural models. *Image and Vision Computing*, pages 790–799, 2014. 5, 6

[38] B. Yang, J. Yan, Z. Lei, and S. Z. Li. Aggregate channel features for multi-view face detection. In *IJCB*, pages 1–8, 2014. 6

[39] S. Yang, P. Luo, C.-C. Loy, and X. Tang. From facial parts responses to face detection: A deep learning approach. In *ICCV*, pages 3676–3684, 2015. 6

[40] S. Yang, P. Luo, C.-C. Loy, and X. Tang. Wider face: A face detection benchmark. In *CVPR*, pages 5525–5533, 2016. 1, 5, 6

[41] S. Yang, Y. Xiong, C. C. Loy, and X. Tang. Face detection through scale-friendly deep convolutional networks. *arXiv preprint arXiv:1706.02863*, 2017. 6

[42] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang. Unitbox: An advanced object detection network. In *ACMMM*, pages 516–520, 2016. 1

[43] C. Zhang, X. Xu, and D. Tu. Face detection using improved faster rcnn. *arXiv preprint arXiv:1802.02142*, 2018. 6

[44] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, pages 1499–1503, 2016. 6

[45] S. Zhang, X. Zhu, Z. Lei, H. Shi, X. Wang, and S. Z. Li. Sˆ 3fd: Single shot scale-invariant face detector. In *ICCV*, pages 192–201, 2017. 1, 2, 3, 6

[46] C. Zhu, R. Tao, K. Luu, and M. Savvides. Seeing small faces from robust anchor's perspective. In *CVPR*, pages 5127–5136, 2018. 1, 2, 6

[47] 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*, pages 57–79. 2017. 6

[48] X. Zhu and D. Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *CVPR*, pages 2879–2886, 2012. 5, 6