Machine Intelligence Research
www.mi-research.net

DOI: 10.1007/s11633-023-1423-y

YuNet: A Tiny Millisecond-level Face Detector

Wei Wu[1] Hanyang Peng[2] Shiqi Yu[1]

[1] Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China

[2] Pengcheng Laboratory, Shenzhen 518066, China

Abstract: Great progress has been made toward accurate face detection in recent years. However, the heavy model and expensive computation costs make it difficult to deploy many detectors on mobile and embedded devices where model size and latency are highly constrained. In this paper, we present a millisecond-level anchor-free face detector, YuNet, which is specifically designed for edge devices. There are several key contributions in improving the efficiency-accuracy trade-off. First, we analyse the influential state-of-the-art face detectors in recent years and summarize the rules to reduce the size of models. Then, a lightweight face detector, YuNet, is introduced. Our detector contains a tiny and efficient feature extraction backbone and a simplified pyramid feature fusion neck. To the best of our knowledge, YuNet has the best trade-off between accuracy and speed. It has only 75 856 parameters and is less than 1/5 of other small-size detectors. In addition, a training strategy is presented for the tiny face detector, and it can effectively train models with the same distribution of the training set. The proposed YuNet achieves 81.1% mAP (single-scale) on the WIDER FACE validation hard track with a high inference efficiency (Intel i7-12700K: 1.6 ms per frame at 320×320). Because of its unique advantages, the repository for YuNet and its predecessors has been popular at GitHub and gained more than 11 K stars at https://github.com/ShiqiYu/libfacedetection

Keywords: Face detection, object detection, computer version, lightweight, inference efficiency, anchor-free mechanism.

Citation: W. Wu, H. Peng, S. Yu. YuNet: A tiny millisecond-level face detector. *Machine Intelligence Research*. http://doi.org/10.1007/s11633-023-1423-y

1 Introduction

Face detection has been an attractive topic in computer vision for decades. It is heavily dependent as a prerequisite step for many face-related applications such as face recognition, face beautification, face alignment, face tracking, etc. Given an image, face detection locates the face regions by bounding boxes. Many methods have been proposed to improve face detection performance, from early hand-crafted features such as Haar in [1] to current CNN-based features. As described in [2], the runtime of the two-stage or multi-stage detectors depends on the number of faces. Therefore, the single-stage CNN-based detectors have become popular in recent years.

Face detection is less challenging than generic object detection. The accuracy reaches saturation on the challenging benchmark WIDER FACE[3]. Some people may think face detection is a solved problem. However, it is not. The top-ranked methods[4−11] all use large pre-trained backbone networks, complex feature enhancement modules and heavy test time augmentations (TTAs) for better ranks[2]. For example, one of the best detectors, Mogface[10], achieves state-of-the-art accuracy with 711 M parameters and 808 GFLOPs (for VGA images). The impressive accuracy comes from the consumption of considerable storage and computation resources.

However, face detection is widely deployed on edge devices such as cell phones, service robots, surveillance cameras and Internet of things (IoT) devices in real-world applications. These devices have limited storage resources and computing capability due to their cost. In addition, only a few noticeable faces need to be detected, and tiny faces in the background are normally not needed in many applications. Even when deployed in a central server, a fast and efficient detector can save considerable energy and make the server handle considerable data synchronously. Compared with a huge face detector that can improve the average precision (AP) slightly on some benchmarks, we argue that an efficient tiny detector is more urgently needed.

The backbone networks in a face detector are essential for performance. Some popular backbone networks such as VGG-16 from the VGGNet[12] series, ResNet-50/101/152 from the ResNet[13] series and MobileNet[14] were originally designed for image classification of ImageNet[15]. As shown in Fig. 1, face detection is different from image classification, which takes the output of the deepest layer as the feature vector. To handle objects of different scales, different feature maps from different layers are employed for detection. Large faces are easier to

Research Article
Manuscript received September 2, 2022; accepted February 6, 2023
Recommended by Associate Editor Ming-Ming Cheng

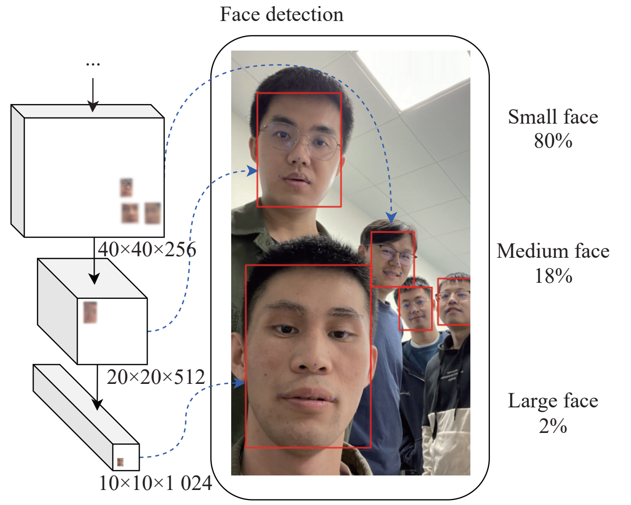


Fig. 1 To handle faces of different sizes, normally large faces will be detected from a deeper feature map, and small faces will be detected from a shallower feature map since a pixel on different feature maps has a different field of view.

detect due to the richness of information. In addition, large faces are normally detected from a deeper feature map and are easier to detect than smaller faces. It gives a strong hint that the backbone should focus on small faces in face detection.

We should also note the distribution of the face sizes. In the WIDER FACE dataset, most faces are small ones, which are less than 20 pixels. It is similar in many face-related applications. Many data augmentation operations, especially random cropping, will change the distributions of face sizes. If we train a model with a dataset of different distributions (distribution A, B and C in Fig. 2), the AP will decrease obviously. The further from the original distribution, the lower AP will be.

A tiny millisecond-level face detector, YuNet, has been designed and presented in the following part of the paper. The contributions of the paper are listed as follows.

1) According to our unique understanding of face detection, we designed a tiny face detector, which has a very limited number of parameters, a very low latency and promising accuracies.

2) We suggested a data sampling strategy for model training. It can obviously improve the accuracy of a deep detector, especially of a lightweight detector.

3) To the best of our knowledge, the proposed YuNet should be the best tiny face detector, which achieves an AP of 81.1% on the WIDER FACE validation hard set and has gained more than 11 K stars at GitHub.com for its effectiveness.

2 Related works

Face detection is a popular topic in object detection and is also very mature for real applications. In the past decade, deep learning-based face detection can handle face scale, pose, occlusion, expression, makeup, illumination, blur, etc., very well. Some benchmarks, such as WIDER FACE[3], have been widely used for evaluating different methods and have promoted research.

As introduced in [2], the latency of a two-stage face detector varies with the number of faces in an image. Single-stage detectors have become more popular in recent years. Some recent single-stage detectors are as follows. Najibi et al.[16] build three detection modules cooperating with context modules for scale-invariant face detection. RetinaFace[9] employs additional facial landmark annotations to improve hard face detection. Li et al.[7] introduce small face supervision signals on the backbone, which implicitly boosts the performance of pyramid features. Zhang et al.[17, 18] adopt neural architecture search (NAS) on feature enhancement modules and face-appropriate necks, respectively, for efficient context enhancement and multiscale feature fusion. Zhang et al.[5], Chi et al.[6], Tang et al.[19], and Liu et al.[20] work on different anchor sampling/matching strategies to balance the proportion of positive and negative samples, match outer faces with high-quality anchors and accelerate model convergence. All these methods achieve extremely high accuracies by employing techniques such as complicated feature enhancement modules, sophisticated anchor

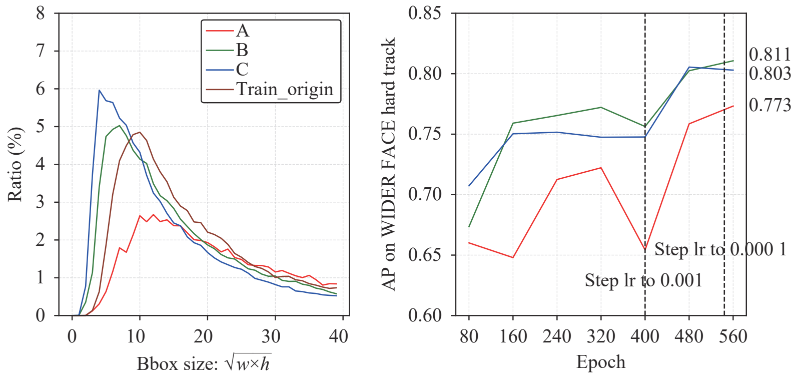


Fig. 2 If we train a face detector with datasets of different distributions (A in red, B in green, and C in blue), the AP tends to decrease as the distribution moves further away from the original distribution.

matching/alignment, and training strategies. The expensive cost and intolerable latency hinder their application in real-world scenarios.

Many efficient face detection methods have been developed to address practical applications. YOLO5Face proposed by Qi et al.[21], inherited from the YOLOv5[22] generic object detector, adds facial key point labels and optimized data augmentation. Guo et al.[23] introduce sample redistribution (SR) to augment training samples and computation redistribution (CR) to reallocate the computation across different components (backbone, neck, and head) and a broad range of computing regimes. Feature fusion is a key technology for improving feature representation, and it is widely used in face detection and some other tasks. For example, gOctConv proposed by Gao et al.[24] fuse both in-stage and cross-stages multiscale features. Those works design models to meet resource constraints. However, we believe the models can be even smaller and faster.

RetinaFace[9] is a recent good detector for face detection and achieves excellent accuracy on the WIDER FACE[3] benchmark. The whole network can be divided into three components, i.e., backbone, neck, and head. The backbone consists of ResNet-50[13] except the adaptive average pooling layer and the fully connected layer, and outputs feature maps of 1/8, 1/16, and 1/32 of the input resolution, respectively. The neck is a standard feature pyramid network (FPN)[25], which consists of a combination of lateral and vertical paths. The head consists of multiple cascaded feature enhancement modules (FEMs) and a convolution for output classification and regression. The RetinaFace model has 27.27 M parameters and 11.07 GFLOPs with an input of size 320 × 320.

3 Methodology

Before introducing the proposed YuNet, some analysis and design principles will be given first. By analysing the relationship among the model size, computational cost and speed, we can have some ideas on how to design a good backbone for face detection. We take RetinaFace[9] as an example to analyse how to design a good detector and then introduce our YuNet in this section. Most CNN-based face detectors follow a similar manner as RetinaFace.

3.1 Analysis on different layers

3.1.1 Number of parameters of different layers

A tiny face detector has many advantages. In addition to its fast inference speed, it is also easy to deploy on many edge devices with limited random access memory (RAM) and limited storage. More parameters can bring a better detection accuracy. However, we have to consider which layers deserve more parameters.

Most parameters in a CNN are in convolutional layers. For standard convolution, the number of parameters is

$$\#Params = K^2 \times C_{in} \times C_{out} \tag{1}$$

where K, C_{in} and C_{out} represent the convolution kernel size, generally 3 or 5, the input and output channels respectively. Obviously, the number of parameters does not depend on the size of the feature map, but is correlated with the size of the convolution kernel and the number of channels.

The numbers of parameters of the convolutional layers in RetinaFace are shown in Fig. 3 as blue bars. The number of parameters increases exponentially, and the deepest layer, Layer 4, contributes 63.55% of the parameters. The reason is that the number of channels increases greatly. It also shows that we should reduce the number of channels of some deep layers if we want to reduce the model size.

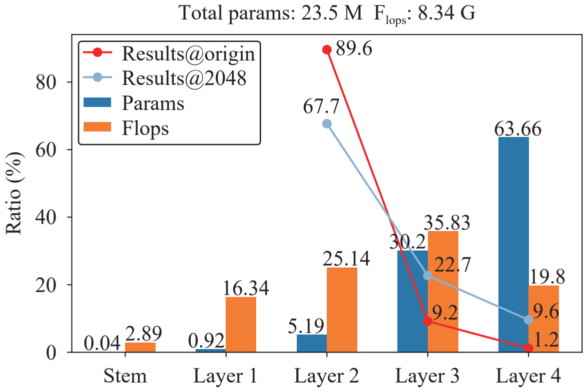


Fig. 3 Numbers of parameters and the computational costs of different convolutional layers in Retinaface's backbone (ResNet-50). The red line and the light blue line indicate the predicted candidates of Layers 2–4 on WIDER FACE under two conditions, which are with original image sizes and with resized images of the long edge to 2048.

3.1.2 Computational cost of different layers

Floating point operations per second (FLOPs) is a widely used measure for computational cost. We can also use it to evaluate different layers. Since the convolutional layers contribute more than 90% of the computational cost of most CNN models, we only consider the FLOPs of the convolutional layers.

$$\begin{aligned}\#FLOPs = H_{out} \times W_{out} \times K^2 \times C_{in} \times C_{out} = \\ H_{out} \times W_{out} \times \#Params\end{aligned} \tag{2}$$

where H_{out} and W_{out} are the height and width of the output feature map respectively.

The computational costs of different layers are plotted as orange bars in Fig. 3. We find that Layer 1 contributes 16.34% of the total computational cost, but only 0.92% of the parameters. Layer 4 contributes 19.8% of the computational cost, but 63.66% of parameters. Fig. 3 shows that the computational cost is not highly correlated with the number of parameters.

3.1.3 Contributions of different layers

From the red line and the light blue line in Fig. 3, Layer 2 contributes more than half of the candidates for face detection, but Laye 4 only contributes less than 10%, even though it has 63.55% of the parameters and 19.8% of the computational cost. The small faces, which will be predicted from Layer 2, should be put more emphasis than those large faces. Large faces are easier to detect due to their rich information, so it is not necessary to have too many channels in Layer 4, or some deeper layers.

3.2 YuNet

According to the previous analysis, we designed a tiny network for face detection. One principle is to focus on difficult small faces and remove computational cost from easy large faces. Another one is to use depthwise convolution and pointwise convolution to replace standard convolution. The architecture of the proposed YuNet is shown in Fig. 4, and it contains a backbone, a tiny feature pyramid network (TFPN) neck and a head.

3.2.1 Backbone

The backbone is the main part of the network and is used for extracting features for detection. It must be efficient and lightweight. To deploy convolutional neural networks to edge devices, efficient units with fewer parameters and faster speed are expected. Depthwise separable convolution is a form of factorized convolution that factorizes the standard convolution into depthwise convolution and pointwise convolution. It is originally from MobileNet[14]. The 3×3 depthwise separable convolution achieves approximately 1/9 to 1/8 computational and parametric costs of the 3×3 standard convolution, and the accuracy may decrease slightly. The DWUnit in YuNet is created by a depthwise separable convolution and its following batch normalization and activation layer, and it is the main module in the proposed network. Another module is DWBlock, which contains two DWUnits. Their designs are presented in the top-right corner of Fig. 4.

The backbone consists of 5 stages. Stage 0, a ConvHead module, contains a standard convolution layer with 3×3 kernels and a stride of 2. Stage 0 is followed by Stage 1, which contains a maxpooling, a ReLU and two DWBlocks. The first two stages, Stage 0 and Stage 1, reduce the feature maps to 1/4 of the input size and increase the channels from 3 to 64. The remaining three stages, Stages 2–4, use exactly the same network structure and feed their hierarchical output feature maps to the TFPN neck.

3.2.2 Neck

The neck is for fusing multiscale features for a higher level of features. In our YuNet, the TFPN neck takes advantage of FPN and depthwise separable convolution. We can consider the feature map from Stage 2 as low-level features and the one from Stage 4 as high-level features because of their different depths. One of the pioneering works, the FPN[25], introduces a top-down pathway and lateral connections to combine multiscale features. The top-down pathway in FPN can generate higher-resolution features by upsampling feature maps from higher pyramid levels. The connections combine feature maps of the same spatial size from the backbone and the top-down pathway. Fig. 5(a) shows how FPN fuses multiscale features. Its fusion can be formulated as follows:

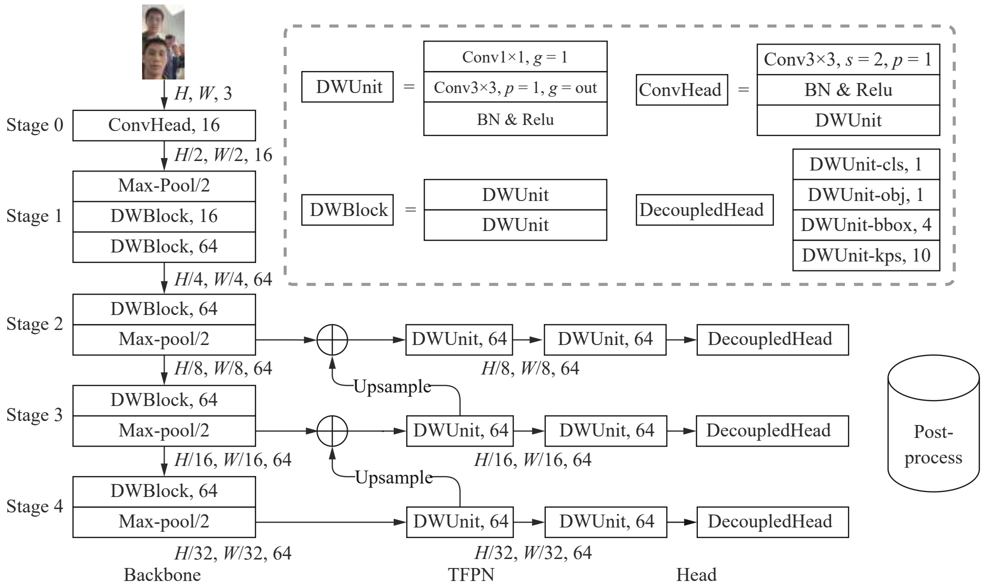


Fig. 4 Architecture of YuNet. It consists of a backbone, a TFPN neck and a head, which are all based on depthwise separable convolution. "ConvHead, 16" indicates a module named ConvHead with 16 output channels. All the head outputs will be concatenated together and produce detection results via non-maximum suppression (NMS).

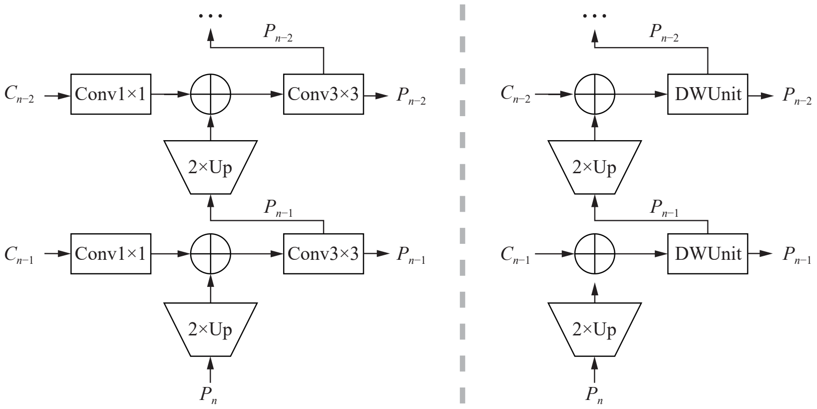


Fig. 5 Structures of (a) FPN and (b) the proposed TFPN

$$P_i = F_i(R_{dim}(C_i) + R_{size}(C_{i+1})) \tag{3}$$

where R_{dim} a channel adjustment operation for dimension matching, R_{size} is an upsampling operation for resolution matching, and F_i is usually a convolutional operation for feature processing. The 1×1 convolution, 2×up and 3×3 convolution in Fig. 5(a) correspond to these three operations.

However, the standard convolutional layers in FPN have a heavy computational cost and a lot of parameters need to be trained. The TFPN neck uses depthwise separable convolution to replace the standard convolution. As shown in Fig. 5(b), DWUnit replaces Conv 3×3 of FPN. DWUnit is the dominant module of the backbone and in the neck. It can reduce the number of parameters to 12% from that of FPN. The computational cost is also greatly reduced.

3.2.3 Head

We adopt the anchor-free mechanism to conduct our YuNet. Compared to the anchor-based RetinaFace, we reduce the candidates for each location from more than 2 to 1 and make them directly predict the four values of a location, which are the two values for the left-top corner, the height and the width of the box. Inspired by Ge et al.[26], we employ simple optimal transport assignment (simOTA)[27] for positive anchor matching. For any matching candidates, we compute the intersection over union (IOU) between the predicted bounding box and ground truth as the soft label. We then use CrossentropyLoss to compute the classification loss. The losses of bounding box regression, landmark regression and objectness are extended-IoULoss (EIOU)[28], SmoothL1Loss and CrossEntropyLoss, respectively. In the training phase, we minimize the multitask loss:

$$L = \frac{1}{N}\sum^{N}(L_{cls} + \alpha_1 L_{obj} + \alpha_2 L_{bbox} + \alpha_3 L_{ldm}) \tag{4}$$

where N indicates the total number of positive samples. The hyper-parameters α_1, α_2 and α_3 are recommended to be 1.0, 5.0 and 0.1, respectively.

4 Model training

4.1 Scale augmentation

Due to the extremely large scale variations (from several pixels to thousands pixels) of faces in real-world scenarios, different scale augmentation strategies are employed to adjust the sample scale distribution in the training phase. The most popular scale augmentation strategies are RandomCrop and its variants. Given an image, a square patch is cropped from the original images, whose size is the short edge product with a scale randomly selected from a predefined set of scales. Then, the patch is resized to 640×640 in case of tensor alignment of an entire batch. The predefined scale set will change the scale distribution of the training samples. A commonly used value range is [0.3, 1.0]. To generate more positive tiny samples, SCRFD proposed by Guo et al.[23] enlarges the random size range to [0.3, 2.0]. When the scale is greater than 1.0, the cropped box will be beyond the original image. The regions out of the range will be filled with the average RGB values. Moreover, they present a searchable zoom-in and zoom-out space to search the optimal range set of scales under the estimation metric of AP on WIDER FACE. With enough rounds of searching, this searchable algorithm can finally obtain the optimal set of scales achieving the best AP on the WIDER FACE validation dataset.

Normally we do not know the face scale distribution in real scenarios, and it is impossible to provide the optimization criteria for the search algorithm. We explicitly indicate that a consistent sample scale distribution between testing and training makes performance optimal. Intuitively, in real-world scenarios, if there are primarily large faces (Easy track of WIDER FACE), we can increase the proportion of small scale of the range set, i.e., [0.3, 0.8]. If there are primarily small-scale faces (Hard track of WIDER FACE), we can increase the proportion of large-scale of the range set, i.e., [0.3, 2.0]. If we want the best average AP on the WIDER FACE, we can keep the sample scale distribution similar between the input training samples and the original training set, i.e., [0.5, 1.5]. It is worth indicating that we need to tune the range set of scales according to the statistics information of only one epoch instead of the entire training phase. With such a simple and intuitive strategy, our model is more accessible for deployment to various real-world scenarios and achieves optimal performance. In addition, we only adopt random horizontal flipping, with a probability of 0.5 besides scale augmentation.

4.2 Training details

To conduct experiments in a more organized manner, we implement the proposed YuNet by PyTorch and opensource MMDetection[29]. We adopt the stochastic gradient descent (SGD) optimizer with a momentum of 0.9, a weight decay of 0.000 5, and a batch size of 16 × 2 on two NVIDIA 1080Ti (12 GB) GPUs. The learning rate is linearly from 0.001 warmed up to 0.01 within the first 1 500 iterations. We adopt the StepLr scheduler to make the learning rate decay by a factor of 10 at the 400th and 544th epochs. Without any pretraining, the model can be well trained from scratch in 560 epochs.

5 Experiments and results

5.1 Dataset

WIDER FACE[3] is the largest public face detection dataset and has 32 203 images and 393 703 faces. The images are split into three subsets for training, validation and testing. Each subset is divided into three levels of difficulty: Easy, Medium, and Hard. Its large variety of scale, pose, occlusion, expression, illumination and event is close to reality and very challenging. Furthermore, we empirically analyse the annotations of the data and observe that Hard covers Medium and Easy, which indicates that performance on Hard can better reflect the effectiveness of different methods. In the following experiments, we pay more attention to performance on Hard.

5.2 Evaluation on WIDER FACE

To make a comprehensive evaluation in terms of model complexity, inference latency, and detection accuracy, we collect some recent methods with similar research purposes according to two requirements: 1) the model size is less than 3 MB, and the computation is less than 1 GFLOPs (640 × 640 input resolution); and 2) the source code has been released, and the trained models have been provided. The following state-of-the-art face detectors are collected for comparison. They are SCRFD[23], RetinaFace[9] and YOLO5Face[21]. Some good detectors, such as BlazeFace[30] and FaceBoxes[31], are not included because they have no officially released source code. The performance results of SCRFD, RetinaFace and YOLO5Face are achieved under different test conditions. For a fair comparison, we do not refer to their experimental results and evaluate all the detectors in our experiments. They have been evaluated as follows.

1) All compared models are converted to ONNX format, and then the inference phase is performed under CPU using ONNXRuntime. We reimplement the preprocessing and postprocessing with NumPy referring to the official code.

2) Without any test time augmentation (TTA) tricks (e.g., image flip, image pyramid, model ensemble, etc.), we use 320 × 320, 640 × 640 and the original size (approximately 1 000 pixels) as the input resolution, to evaluate the models on the WIDER FACE validation subset. We set the confidence threshold close to zero, 0.01, during evaluation to obtain the best mAP, although that will cause a large number of false detections. The NMS threshold is fixed at 0.45.

3) We only evaluate inference efficiency under CPU (Intel i7-12700K) because most mobile and embedded devices only have CPU but no GPU. The evaluation results are obtained by averaging the total inference time spent testing the whole WIDER FACE validation dataset.

The comparisons are listed in Table 1, and the best ones are highlighted. The proposed YuNet has the fewest number of parameters and the lowest computational cost. YuNet has only 75 856 parameters and is almost an order of magnitude smaller than most other small models in the Table 1. According to the latency, YuNet is several times faster than most other models. The accuracy, evaluated by AP_{easy}, AP_{medium} and AP_{hard}, of YuNet is similar to that of other models. In short, YuNet can achieve a similar accuracy to most other small models, but it has much fewer parametres and is much faster.

In addition, we detect the world's largest selfie in Fig. 6, with the input resolution being origin size and the confidence threshold being 0.5. Our YuNet can accurately detect 619 faces out of approximately 1 000 faces reported. Analysis on the face scale reveals that undetected faces contain only a few pixels with blurred or even unrecognizable features to the human eye. In contrast, faces that are obvious and recognizable are accurately located.

5.3 Ablation study

To better understand our YuNet, we further conduct experiments to examine how to impact performance by adding or removing some components and present the comparison in Table 2. Some modules are removed from YuNet (the symbol −) and some modules are strengthened (the symbol +) to discover the functions of different modules. The first row shows a noticeable accuracy

Table 1 Comparison of YuNet with other well-known methods. YOLO5Face[21] does not participate in the comparison of origin size since its ONNX model exported does not support dynamic size input. YuNet-s, RetinaFace and SCRFD-10g are not involved in the comparison of values.

Input size	Methods	#Params (ratio)	#FLOPs (M)	AP_{easy}	AP_{medium}	AP_{hard}	Latency (ms)
320×320	SCRFD-0.5g (ICLR22)[23]	631 410 (8.32×)	195	0.850	0.754	0.372	3.4
	RetinaFace-0.25 (CVPR20)[9]	426 608 (5.62×)	245	0.765	0.611	0.271	4.2
	YOLO5Face-n (Arxiv21)[21]	446 376 (5.88×)	185	0.858	0.793	0.445	7.2
	YuNet (Ours)	75 856 (1.00×)	149	0.836	0.747	0.395	2.2
	YuNet-s (Ours)	54 608 (0.72×)	96	0.785	0.668	0.309	1.9
	RetinaFace	27 293 600 (359.81×)	11 070	0.868	0.742	0.341	49.1
	SCRFD-10g	4 229 905 (55.76×)	3 359	0.923	0.862	0.504	17.3
640 × 640	SCRFD-0.5g	–	779	0.907	0.882	0.684	17.8
	RetinaFace-0.25	–	981	0.893	0.831	0.541	22.0
	YOLO5Face-n	–	741	0.907	0.880	0.734	20.1
	YuNet (Ours)	–	595	0.899	0.869	0.691	11.3
	YuNet-s (Ours)	–	386	0.876	0.834	0.591	8.7
	RetinaFace	–	44260	0.943	0.908	0.659	232.7
	SCRFD-10g	–	13435	0.949	0.935	0.814	95.0
Origin Size	SCRFD-0.5g	–	–	0.892	0.885	0.820	25.0
	RetinaFace-0.25	–	–	0.907	0.883	0.742	57.0
	YuNet (Ours)	–	–	0.892	0.883	0.811	16.3
	YuNet-s (Ours)	–	–	0.887	0.871	0.768	13.8
	RetinaFace	–	–	0.955	0.941	0.847	463.7
	SCRFD-10g	–	–	0.923	0.925	0.885	137.8

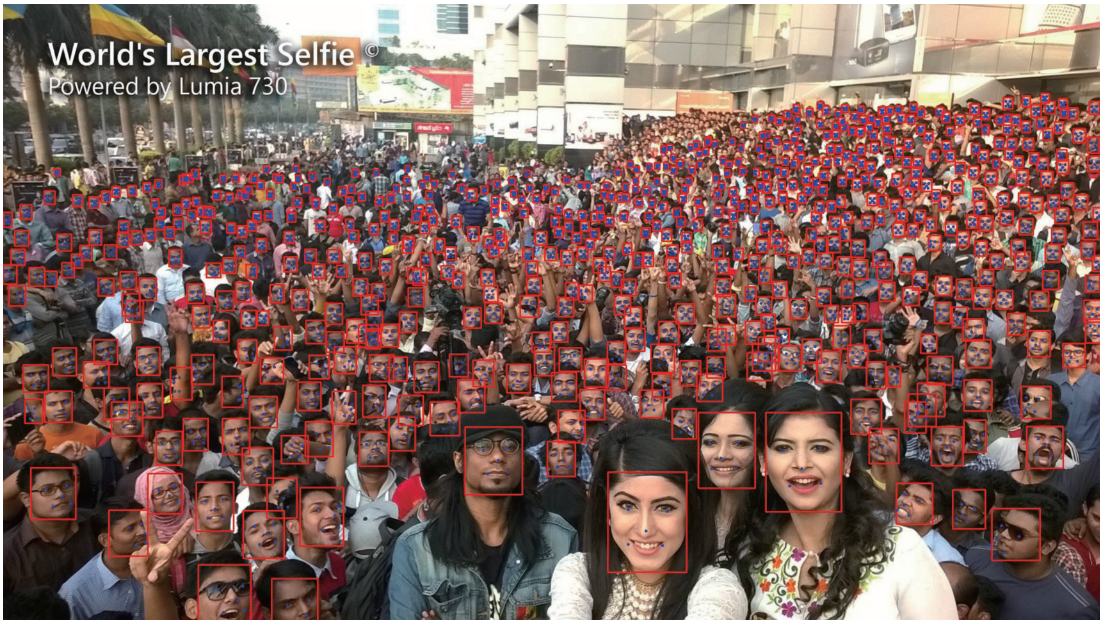


Fig. 6 The world's largest selfie. Our YuNet successfully detects 619 faces out of approximately 1 000 faces reported. The undetected face contains only a few pixels, is blurred and can not be recognized by human eyes. They can be directly ignored in real applications.

drop compared to the others. This indicates that the deep feature maps from Stages 3 and 4 are indispensable even there are few large faces. The second row illustrates that the proposed TFPN neck can boost accuracy by 1–2 per-

Table 2 Ablation study of the proposed modification. In the fourth column, "1" represents that the detection head consists of one DWUnit while "2" consists of two DWUnit, i.e., one DWBlock.

Name	Stage 1 Out channels	Stage 2, 3, 4 Out channels	Head stacked	AP_{easy}	AP_{medium}	AP_{hard}	#MFLOPs	#Params (320 × 320)
YuNet – Top Stage 3, 4	64	[64]	2	0.791 (−0.101)	0.815 (−0.068)	0.744 (−0.067)	138.1	34 032
YuNet – TFPN	64	[64, 64, 64]	2	0.885 (−0.007)	0.871 (−0.012)	0.789 (−0.022)	148.7	75 856
YuNet	64	[64, 64, 64]	2	0.892	0.883	0.811	148.7	75 856
YuNet + 1×1	64	[64, 64, 64]	2	0.890 (−0.002)	0.883	0.812 (+0.001)	157.4	88 336
YuNet + Expand	64	[64, 128, 256]	2	0.906 (+0.014)	0.893 (+0.010)	0.820 (+0.009)	166.8	153 936
YuNet-s	32	[64, 64, 64]	1	0.887 (−0.005)	0.871 (−0.012)	0.768 (−0.043)	96.4	54 608

cent without adding any parameters. The fourth row shows that the 1 × 1 convolution in the FPN serves channel alignment and can be discarded since we have already aligned to 64. We increased the number of channels for Stages 3 and 4 in the fifth row, and the results show that the accuracy is improved by only approximately 1%. In addition, YuNet-s, implemented by halving the number of channels in Stage 1, reduces the computational cost by half, the accuracy drops but not obviously. YuNet-s can be an excellent face detector with a satisfying accuracy for applications that have very limited computational resources.

Another oblation study is on the sample scale distribution. Different data augmentation methods have different distributions. We listed 4 augmentation methods and their scale distributions in Fig. 7. Their corresponding results are listed in Table 3. The best range is [0.5, 1.5], and its scale range (the cyan line in Fig. 7) is the closest to the original range (the red line). Therefore we chose [0.5, 1.5], which has a very close distribution to the original training samples, in our experiments.

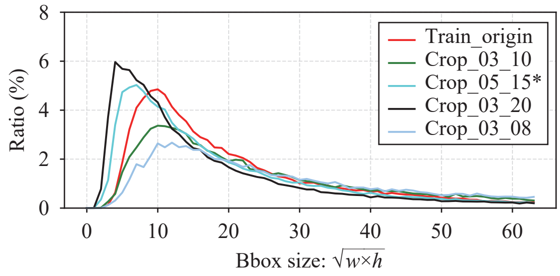


Fig. 7 Vertical axis indicates the ratio for each scale of boxes, and the horizontal axis indicates the specific value.

5.4 Inference efficiency

As presented in Table 1, YuNet demonstrates superior inference efficiency compared to all other detectors across all resolutions, thanks to its compact backbone, innovative TFPN neck, and straightforward detection head.

Table 3 Oblation study of different sample scale distributions.

Range	AP_{easy}	AP_{medium}	AP_{hard}	Average
[0.3, 0.8]	0.902	0.886	0.773	0.854
[0.3, 1.0]	0.901	0.885	0.794	0.860
[0.5, 1.5]*	0.892	0.883	0.811	0.862
[0.3, 2.0]	0.886	0.876	0.803	0.855

* denotes the best performance and is adopted by YuNet and YuNet-s for comparison

We also study speed at other resolutions, and the results are shown in Table 4. The test conditions are consistent with those mentioned in Section 5.2, except for looping through an image 1 000 times instead of the entire validation subset. Our YuNet and YuNet-s can run at considerable real-time speed for all listed resolutions, even for 1280 × 960 images.

Table 4 Inference efficiency of YuNet at various commonly used image sizes. The experiments were carried out with ONNXRuntime on a CPU of Intel i7-12700K. The latency is the average time of a loop of 1 000.

Resolution	YuNet		YuNet-s	
	Latency (ms)	FPS	Latency (ms)	FPS
224 × 224	1.1	909	0.9	1111
320 × 320	1.6	625	1.4	714
640 × 480	9.6	104	7.8	128
1280 × 960	36.2	28	29.3	34

6 Conclusions

In this paper, an efficient tiny face detector, YuNet, is specifically designed for real-time applications. It can achieve a millisecond-level speed on CPUs, and is suitable for mobile and embedded devices. The design of YuNet is inspired by the principles for efficient small models. We studied different components and strategies (backbone, neck, head, training, etc.) of face detection to make a good trade-off between the computation cost and the accuracy. To the best of our knowledge, YuNet should be the smallest and simplest model for face detection. In the future, we hope to continue to reduce the size of the model and to improve the speed while keeping the accuracy unchanged.

Acknowledgements

This work was supported in part by National Natural Science Foundation of China (No. 61976144), the Stable Support Plan Program of Shenzhen Natural Science Fund (No. 20200925155017002), and the National Key Research and Development Program of China (No. 2020 AAA0140000).

Open Access

References

[1] P. Viola, M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Kauai, USA, pp. 511–518, 2001. DOI: 10.1109/CVPR.2001.990517.

[2] Y. T. Feng, S. Q. Yu, H. Y. Peng, Y. R. Li, J. G. Zhang. Detect faces efficiently: A survey and evaluations. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, vol. 4, no. 1, pp. 1–18, 2021. DOI: 10.1109/tbiom.2021.3120412.

[3] S. Yang, P. Luo, C. C. Loy, X. O. Tang. WIDER FACE: A face detection benchmark. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, USA, pp. 5525–5533, 2016. DOI: 10.1109/cvpr.2016.596.

[4] P. Y. Hu, D. Ramanan. Finding tiny faces. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, USA, pp. 1522–1530, 2017. DOI: 10.1109/cvpr.2017.166.

[5] S. F. Zhang, X. Y. Zhu, Z. Lei, H. L. Shi, X. B. Wang, S. Z. Li. S.3FD: Single shot scale-invariant face detector. In *Proceedings of IEEE International Conference on Computer Vision*, Venice, Italy, pp. 192–201, 2017. DOI: 10.1109/iccv.2017.30.

[6] C. Chi, S. F. Zhang, J. L. Xing, Z. Lei, S. Z. Li, X. D. Zou. Selective refinement network for high performance face detection. *In Proceedings of AAAI Conference on Artificial Intelligence*, vol. 33, no. 1, pp. 8231–8238, 2019. DOI: 10.1609/aaai.v33i01.33018231.

[7] J. Li, Y. B. Wang, C. A. Wang, Y. Tai, J. J. Qian, J. Yang, C. J. Wang, J. L. Li, F. Y. Huang. DSFD: Dual shot face detector. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Long Beach, USA, pp. 5055–5064, 2019. DOI: 10.1109/cvpr.2019.00520.

[8] W. Liu, S. C. Liao, W. Q. Ren, W. D. Hu, Y. N. Yu. High-level semantic feature detection: A new perspective for pedestrian detection. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Long Beach, USA, pp. 5187–5196, 2019. DOI: 10.1109/cvpr.2019.00533.

[9] J. K. Deng, J. Guo, E. Ververas, I. Kotsia, S. Zafeiriou. RetinaFace: Single-shot multi-level face localisation in the wild. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Seattle, USA, pp. 5202–5211, 2020. DOI: 10.1109/cvpr42600.2020.00525.

[10] Y. Liu, F. Wang, J. K. Deng, Z. P. Zhou, B. Sun, H. Li. MogFace: Towards a deeper appreciation on face detection. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, New Orleans, USA, pp. 4083–4092, 2022. DOI: 10.1109/CVPR52688.2022.00406.

[11] L. Song, J. F. Yang, Q. Z. Shang, M. A. Li. Dense face network: A dense face detector based on global context and visual attention mechanism. *Machine Intelligence Research*, vol. 19, no. 3, pp. 247–256, 2022. DOI: 10.1007/s11633-022-1327-2.

[12] K. Simonyan, A. Zisserman. Very deep convolutional networks for large-scale image recognition. [Online], Available: https://arxiv.org/abs/1409.1556, 2014.

[13] K. M. He, X. Y. Zhang, S. Q. Ren, J. Sun. Deep residual learning for image recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, USA, pp. 770–778, 2016. DOI: 10.1109/cvpr.2016.90.

[14] A. G. Howard, M. L. Zhu, B. Chen, D. Kalenichenko, W. J. Wang, T. Weyand, M. Andreetto, H. Adam. MobileNets: Efficient convolutional neural networks for mobile vision applications. [Online], Available: https://arxiv.org/abs/1704.04861, 2017.

[15] A. Krizhevsky, I. Sutskever, G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems*, Lake Tahoe, USA, pp. 1097–1105, 2012.

[16] M. Najibi, P. Samangouei, R. Chellappa, L. S. Davis. SSH: Single stage headless face detector. In *Proceedings of International Conference on Computer Vision*, Venice, Italy, pp. 4885–4894, 2017. DOI: 10.1109/iccv.2017.522.

[17] J. Li, B. Zhang, Y. B. Wang, Y. Tai, Z. Y. Zhang, C. J. Wang, J. L. Li, X. M. Huang, Y. L. Xia. ASFD: Automat-

ic and scalable face detector. In *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 2139–2147, 2021. DOI: 10.1145/3474085.3475372.

[18] Y. Liu, X. Tang. BFBox: Searching face-appropriate backbone and feature pyramid network for face detector. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Seattle, USA, pp. 13565–13574, 2020. DOI: 10.1109/cvpr42600.2020.01358.

[19] X. Tang, D. K. Du, Z. Q. He, J. T. Liu. PyramidBox: A context-assisted single shot face detector. In *Proceedings of the 15th European Conference on Computer Vision*, Springer, Munich, Germany, pp. 812–828, 2018. DOI: 10.1007/978-3-030-01240-3_49.

[20] Y. Liu, X. Tang, J. Y. Han, J. T. Liu, D. E. Rui, X. Wu. HAMBox: Delving into mining high-quality anchors on face detection. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Seattle, USA, pp. 13043–13051, 2020. DOI: 10.1109/cvpr42600.2020.01306.

[21] D. L. Qi, W. J. Tan, Q. Yao, J. F. Liu. YOLO5Face: Why reinventing a face detector. In *Proceedings of Computer Vision – ECCV Workshops*, Springer, Tel Aviv, Israel, vol. 13805, pp. 288–244, 2022. DOI: 10.1007/978-3-031-25072-9_15.

[22] G. Jocher. YOLOv5, 2020. [Online], Available: https://github.com/ultralytics/yolov5, Mar. 2022.

[23] J. Guo, J. K. Deng, A. Lattas, S. Zafeiriou. Sample and computation redistribution for efficient face detection. In *Proceedings of the 10th International Conference on Learning Representations*, 2022.

[24] S. H. Gao, Y. Q. Tan, M. M. Cheng, C. Z. Lu, Y. P. Chen, S. C. Yan. Highly efficient salient object detection with 100K parameters. In *Proceedings of the 16th European Conference on Computer Vision*, Springer, Glasgow, UK, 2020, pp. 702–721. DOI: 10.1007/978-3-030-58539-6_42.

[25] T. Y. Lin, P. Dollár, R. Girshick, K. M. He, B. Hariharan, S. Belongie. Feature pyramid networks for object detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, USA, pp. 936–944, 2017. DOI: 10.1109/cvpr.2017.106.

[26] Z. Ge, S. T. Liu, F. Wang, Z. M. Li, J. Sun. YOLOX: Exceeding YOLO series in 2021. [Online], Available: https://arxiv.org/abs/2107.08430, 2021.

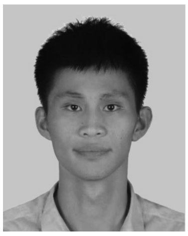
[27] Z. Ge, S. T. Liu, Z. M. Li, O. Yoshie, J. Sun. OTA: Optimal transport assignment for object detection. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Nashville, USA, pp. 303–312, 2021. DOI: 10.1109/CVPR46437.2021.00037.

[28] H. Y. Peng, S. Q. Yu. A systematic IoU-related method: Beyond simplified regression for better localization. *IEEE Transactions on Image Processing*, vol. 30, pp. 5032–5044, 2021. DOI: 10.1109/TIP.2021.3077144.

[29] K. Chen, J. Q. Wang, J. M. Pang, Y. H. Cao, Y. Xiong, X. X. Li, S. Y. Sun, W. S. Feng, Z. W. Liu, J. R. Xu, Z. Zhang, D. Z. Cheng, C. C. Zhu, T. H. Cheng, Q. J. Zhao, B. Y. Li, X. Lu, R. Zhu, Y. Wu, J. F. Dai, J. D. Wang, J. P. Shi, W. L. Ouyang, C. C. Loy, D. H. Lin. MMDetection: Open MMLab detection toolbox and benchmark. [Online], Available: https://arxiv.org/abs/1906.07155, 2019.

[30] V. Bazarevsky, Y. Kartynnik, A. Vakunov, K. Raveendran, M. Grundmann. BlazeFace: Sub-millisecond neural face detection on mobile GPUs. [Online], Available: https://arxiv.org/abs/1907.05047, 2019.

[31] S. F. Zhang, X. Y. Zhu, Z. Lei, H. L. Shi, X. B. Wang, S. Z. Li. FaceBoxes: A CPU real-time face detector with high accuracy. In *Proceedings of IEEE International Joint Conference on Biometrics*, Denver, USA, 2017. DOI: 10.1109/BTAS.2017.8272675.



Wei Wu received the B. Sc. degree in computer science and technology from Chongqing University, China in 2017. Currently, he is a master student in electronics science and technology at Department of Computer Science and Engineering, Southern University of Science and Technology, China.

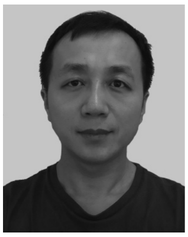
His research interests include object detection and computer vision.

E-mail: 12032501@mail.sustech.edu.cn
ORCID iD: 0000-0002-9595-1778

Hanyang Peng received the B. Sc. degree in measurement and control technology from Northeast University of China, China in 2008, the M. Eng. degree in detection technology and automatic equipment from Tianjin University, China in 2010, and the Ph. D. degree in pattern recognition and intelligence systems from Institute of Automation, Chinese Academy of Sciences, China in 2017. He currently works as an assistant professor in Pengcheng Laboratory, China.

His research interests include computer vision, machine learning and distributed learning.

E-mail: penghy@pcl.ac.cn
ORCID iD: 0000-0002-9715-473X



Shiqi Yu received the B. Eng. degree in computer science and engineering from Chu Kochen Honors College, Zhejiang University, China in 2002, and the Ph. D. degree in pattern recognition and the intelligent systems from Institute of Automation, Chinese Academy of Sciences, China in 2007. He is currently an associate professor in Department of Computer Science and Engineering, Southern University of Science and Technology, China. He worked as an assistant professor and an associate professor in Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China from 2007 to 2010, and as an associate professor in Shenzhen University, China from 2010 to 2019.

His research interests include gait recognition, face detection and computer vision.

E-mail: yusq@sustech.edu.cn (Corresponding author)
ORCID iD: 0000-0002-5213-5877