Hindawi
Scientific Programming
Volume 2022, Article ID 3856424, 11 pages
https://doi.org/10.1155/2022/3856424

Research Article

An Efficient Anchor-Free Face Detector with Attention Mechanisms

Xiangxian Zhu[1,2] **and Yilun Lou**[1,2]

[1]*Department of HMI Research and Development, Ningbo Preh Joyson Automotive Electronics Co., Ltd., Ningbo 315000, China*
[2]*Zhejiang Key Laboratory of Automotive Electronics Intelligence, Ningbo 315000, China*

Correspondence should be addressed to Xiangxian Zhu; steven.zhu@preh.cn

Received 18 March 2022; Revised 3 June 2022; Accepted 7 June 2022; Published 28 June 2022

Academic Editor: Guokai Zhang

Enormous progress has been made in face detection tasks due to the rapid development of deep learning techniques. Meanwhile, debates arise on whether face detection should be treated as a generic object detection task or considered differently. In this paper, we design an efficient anchor-free face detector that focuses on a low flops regime and combines recent advances in generic object detection with the methods for detecting tiny faces. Specifically, we adopt the anchor-free Fully Convolutional One-Stage (FCOS) method with a recently developed Visual Attention Network (VAN) as the base detector. In accordance with the characteristics of the face dataset, we reallocate the computation across the network components by adjusting the network configurations of the base detector. Then we redesign the criteria for marking positive samples to realize a balanced distribution in pixel maps, and we also adopt the quadruple pixel prediction, which enables more positive samples matched with the model outputs. Under VGA resolution, our face detector achieves 70.5% in AP on the hard subset of the WIDER FACE dataset, while the computational cost is only 1.05 Gflops. This accuracy efficiency trade-off is comparable to state-of-the-art results.

1. Introduction

Face detection, as the upstream task of face tracking [1], face alignment [2], and face verification [3], has received significant attention in the computer vision community. Moreover, its accuracy has been boosted by a large margin due to the emergence of deep learning techniques. In the literature, there are debates over whether face detection differs from generic object detection and requires extra effort to improve performance. On the one hand, TinaFace [4] bridges the gap between generic object and face detection by introducing a simple baseline method. Based on existing general modules, TinaFace achieves the state-of-the-art performance of 92.4% in AP on the WIDER FACE dataset [5]. On the other hand, Guo et al. [6] propose the Sample and Computation Redistribution for efficient Face Detection (SCRFD) method and argue that the characteristics of the face dataset should be considered. They believe the optimal design can only be gained by reconfiguring the whole network structure from backbone to head. In particular, SCRFD algorithm redistributes not only training samples but also computation within the network. As a result, the efficient SCRFD outperforms TinaFace when testing under VGA resolution (i.e., 640×480). At the same time, other approaches like additional branches for outputting extra face landmarks in RetinaFace [7] and compensation outer faces to match high-quality anchors in HAMBox [8] help increase the detection performance.

This paper takes ideas from both sides and proposes an efficient anchor-free face detector based on FCOS [9]. This design focuses on a low compute regime (around 1 Gflops) and works under VGA resolution. More specifically, we first take advantage of attention mechanisms and employ VAN [10] as the network backbone. And we use this visual attention backbone that includes Large Kernel Attention (LKA) modules to replace the ResNet [11] backbone in the original FCOS design. We make this modification because a basic block in VAN with the LKA module captures long-range dependency and thus has a better capability of selecting important features. Secondly, network configurations are selected in a way that more computation is reallocated to the shallow stages of the backbone. Experimental results validate this reallocation design principle. We then adjust the responsible range for different pixel maps to fit the small-sized input image. Finally, we propose a quadruple pixel prediction method that produces four bounding box predictions at a single pixel location, which boosts performance with an additional but small computational overhead. To summarize, our main contributions are the following:

(i) Incorporating attention mechanisms into the detector backbone.

(ii) Reallocating computation distribution within the network and redesigning the positive sample matching criteria according to the characteristics of the face dataset.

(iii) Utilizing quadruple pixel prediction to enable the detector to produce more predictions.

We organize the remaining paper as follows. Section 2 gives the related works on face detection, anchor-free detectors, attention mechanisms, and network design spaces. Section 3 describes the proposed efficient anchor-free detector along with our main contributions. Section 4 provides experimental results and analyses. Section 5 concludes the paper with limitations and future work.

2. Related Works

2.1. Face Detection. Detecting faces in an image has received continuous attention in the computer vision community. Before deep learning techniques were involved in face detection, traditional methods [12, 13] were mainly boosting-based algorithms and relied on manually designed features. With the power of deep neural networks, features are automatically extracted given a large amount of training data. The main challenge in face detection is the unconstrained conditions where faces can be occluded or dimly illuminated or have extreme poses and tiny scales. Nowadays the most commonly used benchmark for unconstrained face detection is the WIDER FACE dataset [5], on which many recently developed CNN-based face detectors report their results. Among them, RetinaFace [7] utilizes five extra key points on a face to advance training. TinaFace [4] considers the face detection task as a generic object detection problem and combines existing modules and techniques to achieve state-of-the-art performance. HAMBox [8] and MogFace [14] use different online anchor mining strategies to compensate for outer faces or improve label assignment. On the hard subset of the WIDER FACE dataset, these state-of-the-art algorithms all exceed AP 91.0%. However, high performance comes at the cost of heavy computation. The above face detectors either adopt a multiscale testing method or employ heavy backbones. As SCRFD [6] points out, TinaFace introduces more than 40 Tflops due to its multiscale testing strategy. Even when tested under the single scale of 640 × 640, TinaFace consumes over 100 Gflops. At the same time, its performance drops to AP 81.4%.

Therefore, another challenge in face detection is the trade-off between the detection accuracy and the computational complexity. Due to the nature of CNN-based face detectors, the computational complexity can be reduced directly by shrinking the input image to a smaller size, e.g., VGA resolution. The price for this low computation is the reduction in the accuracy. There have been algorithms that consider low-resolution inputs. In particular, RefineFace [15] measures its speed under VGA resolution but provides test results under the multiscale testing strategy. OS-LFFD [16] proposes an ommateum structure with shared parameters to shrink the model size and reports its results under single inference on the original schema. BlazeFace [17], with its focus on mobile applications, takes the input image at the size of 192 × 192 to reduce computational costs. SCRFD, specially designed for the VGA resolution input, provides a family of face detectors with flops ranging from 0.5 G to 34 G. This family of models are sampled from network design spaces with the design rule that lower stages of the backbone should have larger computation resources than other network components. The proposed detector in this paper focuses on the low compute regime (1 Gflops) and validates itself under the 640 × 480 input size. Moreover, the cumbersome work of designing and matching anchors is eliminated due to its anchor-free nature.

2.2. Anchor-Free Detectors. Mainstream object detectors such as Faster-RCNN [18], SSD [19], YOLOv2, and YOLOv3 [20, 21] predict offsets to predefined anchor boxes to get final bounding boxes. Thus, they are categorized into anchor-based methods. Meanwhile, anchor-free methods have recently gained substantial attention due to their simplicity. For example, CornerNet [22] treats object detection as a keypoint detection problem and predicts a pair of keypoints, i.e., the top-left and bottom-right corners of an object's bounding box. CenterNet [23] goes a little further by adding another center keypoint to detect, improving both precision and recall.

ObjectsAsPoints [24] predicts a keypoint heatmap and local offset features at stride 4. The top 100 peaks in that output keypoint heatmap are the detected object centers. The bounding box predictions are obtained by combining peak locations and corresponding local features. FCOS [9] has similar local offset regression targets as ObjectsAsPoints, with differences in three aspects. First, FCOS marks a pixel in the pixel map as a positive sample when it locates in any ground truth box. ObjectsAsPoints, on the other hand, spreads object center keypoints to a heatmap by a Gaussian kernel. Secondly, five different pixel map levels in FCOS with strides 8, 16, 32, 64, and 128 are used to detect objects of various sizes. Furthermore, FCOS employs an additional centerness branch indicating the relative distance between the pixel location and the predicted bounding box center.

Since there are many of tiny faces to be detected, the multilevel FCOS is adopted as our base anchor-free face detector.

2.3. Attention Mechanisms. Recent years have witnessed the success of attention mechanisms [25, 26]. While initially designed for natural language processing, attention mechanisms have been widely adopted in computer vision tasks, from image classification and object detection to instance segmentation [27–32]. This new adoption brings substantial performance boosts, and the attention-based algorithms dominate nearly all leaderboards in computer vision tasks. The main idea is that attention mechanisms work as an adaptive process of selecting input features. An attention map is produced by this process, and according to the map, essential features are selected.

Attention mechanisms in computer vision can be categorized into four basic categories [33], i.e., channel attention [34], spatial attention [35], temporal attention [36], and branch attention [37]. Self-attention-based vision transformer [27] and its successors [28–32] capture global information by using spatial attention. However, while long-range dependence is captured by self-attention, the computational costs become vast when dealing with a sizeable 2D input. Inspired by MobileNets [38–40], the LKA module [10] utilizes depth-wise convolution, dilated depth-wise convolution, and point-wise convolution to overcome this shortcoming. Local contextual information, long-range dependence, and adaptability are all considered by this simple design. We use the LKA-based Visual Attention Network as our detector backbone.

2.4. Network Design Spaces. In the pioneer works of Radosavovic et al. [41], a new network design paradigm is proposed. Instead of designing the convolutional neural network on the instance level, they try to find sound design principles that can be generalized to a population of networks. This is achieved by parameterizing the network. Network configurations, such as the number of blocks per stage, block width, and bottleneck ratio for each block, are parameters of the network. While the above configurations have limited ranges, their combination has around 10^{18} possibilities [41], which form the original unconstrained network design space. Hundreds of network configuration samples that meet a predefined flop regime are taken out as representative of this considerable design network space. They then train and test each sample configuration.

The analysis of these produced results reveals design principles. For example, consistent bottleneck ratios across stages do not affect model performance. Increasing widths towards the deeper stage tend to perform better. These design principles shrink the original large design space to a smaller one. Meanwhile, new network configurations are sampled within the shrunk design space, and new trends can be observed and become design principles. This shrinking process goes iteratively. While Radosavovic et al. apply the paradigm to the image classification task, SCRFD [6] uses the same method for face detection problems. Networks for classification consist of only the backbone, whereas the detection network needs additional neck and head structures. SCRFD combines the neck and head network configurations into design spaces and then trains and validates configurations on the WIDER FACE dataset. Due to the existence of many tiny-sized faces, the design principle for face detection learned from WIDER FACE is that more computation should be allocated to the early stage of the network where tiny face detection occurs, and the computation of backbone, neck, and head should be jointly adjusted. Inspired by the above works, our anchor-free face directly applies the gained knowledge in SCRFD to the self-attention-based visual attention backbone, feature pyramid neck, and FCOS head.

3. Our Proposed Anchor-Free Face Detector

This section first demonstrates the structure of the proposed anchor-free face detector and its network configurations. Then, we introduce the changes to the network configurations as well as the positive sample marking criteria that better fit the face dataset. Finally, we describe the quadruple pixel prediction method.

3.1. VAN-Based FCOS. Our face detector consists of three components, the backbone, the neck, and the head, as shown in Figure 1. Backbone feature maps from VAN are fed into the Feature Pyramid Network (FPN) [42] that generates neck features. FCOS head takes the neck features and produces several pixel maps that are responsible for different sizes of face bounding boxes.

3.1.1. Visual Attention Backbone. The novel LKA module adopted by the visual attention backbone is the key to achieving state-of-the-art performance [10]. As shown in Figure 2, the LKA module generates an attention map by three successive convolutions with different types. The first is a 5×5 depth-wise convolution (DW Conv) that captures local feature information within the same channel. Then a 7×7 depth-wise dilation convolution (DW-D Conv) finds the long-range dependence spatially still within the same channel. Lastly, a 1×1 point-wise convolution (PW Conv) fuses information across channels. This channel convolution provides the missing channel adaptability, which is not considered in the depth-wise convolution and the depth-wise dilation convolution. The produced attention map represents the importance of features at each spatial and channel location. A high value in the attention map means the feature at the corresponding location is important. When multiplying the attention map with the input feature elementwise, discriminative features are preserved, and noisy features are suppressed. The process of attention map generation can be viewed as a decomposition of large kernel convolution, whereas the considerable computation costs required by large kernel convolution are alleviated.

The overall architecture of the VAN is straightforward. Figure 3(a) shows that it consists of four stages. At the beginning of each stage, an Overlap Patch Embed (OPE)

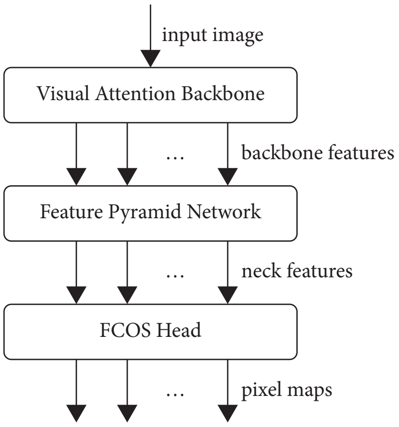


Figure 1: The overall structure of the proposed face detector.

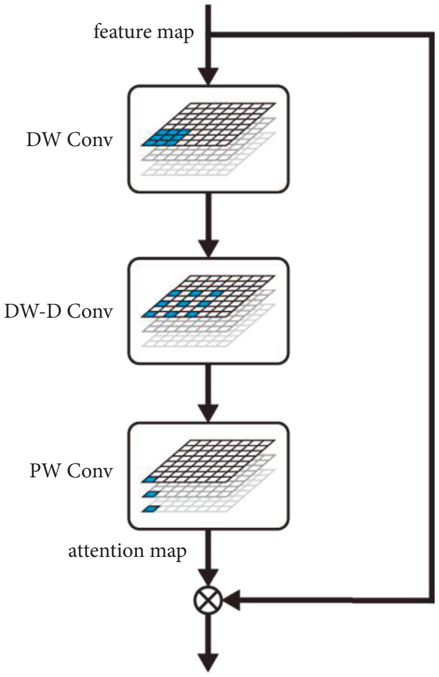


Figure 2: An illustration of Large Kernel Attention module.

module is inserted to downsample the input feature at stride 2 or 4, depending on the stage location. Moreover, there is a sequence of identical VAN blocks within each stage, as shown in Figure 3(b). The length of the sequence in each stage is one network configuration that can be tuned. Finally, a layer normalization layer ends each stage. Output channel sizes for VAN blocks are consistent within stages but may vary across different stages. Consequently, we have four output channel sizes, which are also network configurations. Taking a close look at a VAN block in Figure 3(c), the input feature takes three paths. The middle path is the identity path directly added to the output feature. The left path is the spatial attention path employing the LKA module described ahead. The remaining path uses a multilayer perceptron (MLP) module consisting of a series of point-wise and depth-wise convolutions, as illustrated in Figure 3(c). In an MLP, a hidden channel size is used across convolutions and is defined by multiplying the output channel size of the block and an MLP ratio. MLP ratios are also network configurations. By tuning the MLP ratio, we can easily configure a basic VAN block to a bottleneck or an inverted bottleneck structure. With equivalent large kernel convolution and customizable MLP ratio, the visual attention backbone can maintain the same representation power with fewer parameters and flops compared to the ResNet backbone.

Guo et al. [10] provide a family of van backbones, i.e., VAN-Tiny, VAN-Small, VAN-Base, and VAN-Large. Their network configurations are listed in Table 1. We also add model sizes and flops of the VAN variants when fed with a VGA resolution image. Since we are designing an efficient face detector in a low flop regime, even the tiny version of VAN consumes a large number of computational costs. A simple option to shrink the model is downscaling all output channel sizes by the same factor. We choose 4 as the scaling factor and 0.25 as the MLP ratio. Empirically, the first-stage output channel size is set to 16 to capture enough features. Combined with the above modifications, we term the new set of backbone network configurations as VAN-Reduce whose backbone has low flops.

3.1.2. *FPN and FCOS Head.*

We use the same FPN [42] as that in the original FCOS to acquire high-level semantic feature maps at different levels. The feature map number and the output channel number are also network configurations. In original FCOS head, it includes four standard convolutions before the final prediction layer. Since the detector only needs to detect faces rather than multiple different objects, we manually reduce to only one convolution layer to keep the model compact.

FCOS head produces predictions in a per-pixel way. For the simplicity of illustration, we assume one output pixel map with stride S produced by FCOS head. As shown in Figure 4, the output pixel map has a spatial size of W/S and H/S, where W and H are the width and height of the input image.

A $(4+1+1)-d$ vector at each pixel location contains the distances from pixel location to four boundaries of a bounding box $d=(l,t,r,b)$ the face classification score p, and one centerness score c. A pixel location is indexed by (x_i, y_i), a tuple of two integers. If a pixel at (x_i, y_i) falls into the bounding box of a face in the original image, we mark it as a positive sample and set a label $t^*=1$. In Figure 4, the magenta pixel is a positive sample because its corresponding location in the input image is within the bounding box. Four magenta arrows are the four regression targets,

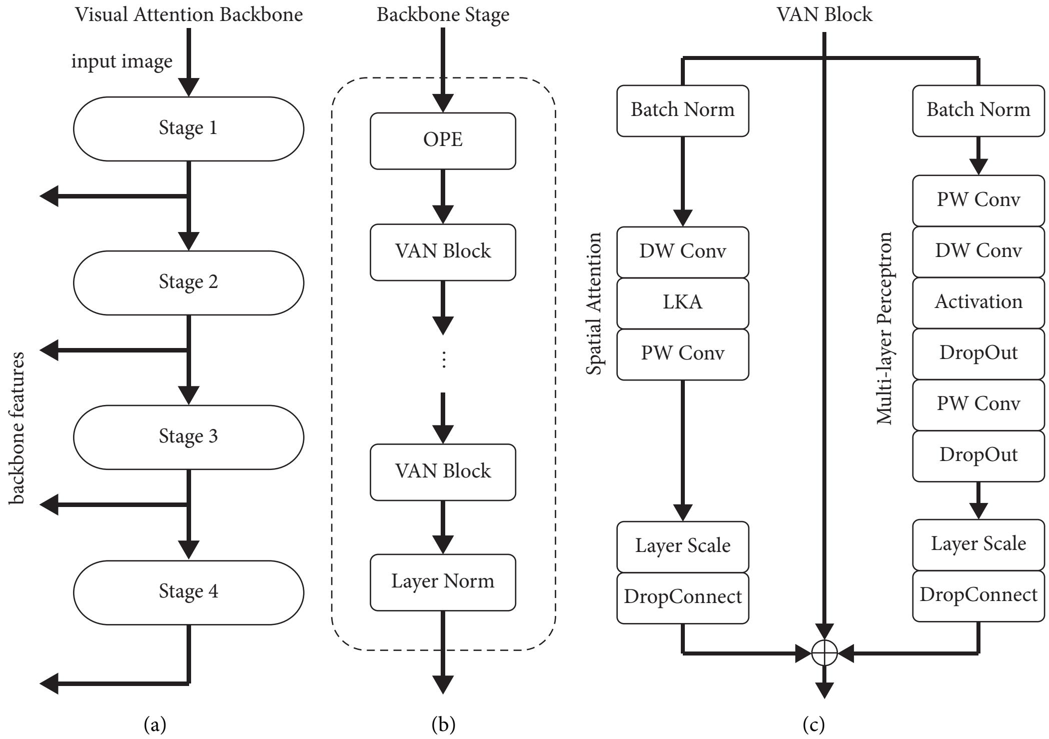


Figure 3: Illustrations of (a) visual attention backbone, (b) a backbone stage, and (c) a VAN block.

Table 1: VAN backbone configurations.

Backbone config.	Numbers of blocks	Output channels	MLP ratios	Flops (G)	Params (M)
VAN-large	3, 5, 27, 3	64, 128, 320, 512	8, 8, 4, 4	55.0	44.3
VAN-base	3, 3, 12, 3	64, 128, 320, 512	8, 8, 4, 4	33.2	27.3
VAN-small	2, 2, 4, 2	64, 128, 320, 512	8, 8, 4, 4	15.4	13.7
VAN-tiny	3, 3, 5, 2	32.64.160, 256	8, 8, 4, 4	5.4	3.9
VAN-reduce (ours)	3, 3, 5, 2	16, 16, 40, 64	0.25, 0.25, 0.25, 0.25	0.3	0.1
VAN-realloc (ours)	3, 5, 3, 2	24, 48, 48, 80	0.25, 0.25, 0.25, 0.25	0.8	0.2

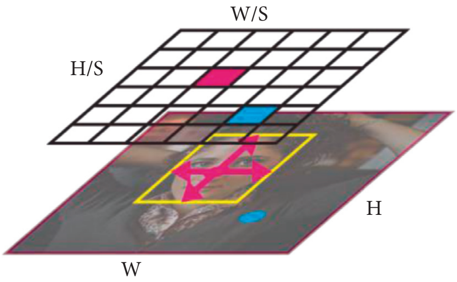


Figure 4: An example of sample matching in FCOS.

$d^* = (l^*, t^*, r^*, b^*)$. If a pixel locates within more than one bounding box, the minimum distance is used as the regression target. By contrast, the cyan pixel is a negative sample ($t^* = 0$). We can recover the ground truth bounding box at the positive sample location by the following formulas:

$$
\begin{aligned}
l_{\text{box}} &= x_c - l^*,\\
t_{\text{box}} &= y_c - t^*,\\
r_{\text{box}} &= x_c + r^*,\\
b_{\text{box}} &= y_c + b^*,\\
x_c &= x_i S + \frac{S}{2},\\
y_c &= x_i S + \frac{S}{2},
\end{aligned}
\tag{1}
$$

where l_{box}, t_{box}, r_{box}, and b_{box} are the left, top, right, and bottom boundaries of the ground truth box; l^*, t^*, r^*, b^* are the regression targets at the pixel location. x_c, y_c denote the coordinates of the pixel center. The centerness score c^* is used to indicate a predicted high-quality bounding box and is defined as follows:

$$c^* = \sqrt{\frac{\min(l^*, r^*)}{\max(l^*, r^*)} \times \frac{\min(t^*, b^*)}{\max(t^*, b^*)}} \tag{2}$$

During inference, the overall score of a prediction is the product of the face classification score p and the centerness score c. A centerness score close to 1 means the pixel center x_c, y_c is near the bounding box center, and the prediction should be considered high-quality.

The loss function, which is the same as that in the original FCOS, is given below:

$$
\begin{aligned}
L\left(\left\{p_{xi,yi}\right\}, \left\{d_{xi,yi}\right\}, \left\{c_{xi,yi}\right\}\right) = {} & \frac{1}{N_{\text{pos}}} \sum_{xi,yi} L_{\text{cls}}\left(p_{xi,yi}, t^*_{xi,yi}\right) +\\
& \frac{1}{N_{\text{pos}}} \sum_{xi,yi} t^*_{xi,yi} L_{\text{reg}}\left(d_{xi,yi}, d^*_{xi,yi}\right)\\
& + \frac{1}{N_{\text{pos}}} \sum_{xi,yi} t^*_{xi,yi} L_{cnt}\left(c_{xi,yi}, c^*_{xi,yi}\right),
\end{aligned}
\tag{3}
$$

where the classification loss L_{cls}, the regression loss L_{reg}, and the centerness loss L_{cnt} are focal loss [43], GIOU loss [44], and binary cross entropy loss, respectively. Since we only have one class to detect, the positive sample label $t^*_{xi,yi}$ is also the class label and is used in the classification loss.

FCOS uses multilevel pixel maps to detect bounding boxes for large-scale variances. In the original design of FCOS, there are five level pixel maps with strides 8, 16, 32, 64, and 128, respectively. We define the above five pixel maps as P8, P16, P32, P64, and P128, together with the corresponding neck outputs as N8, N16, N32, N64, and N128. In this scenario, marking positive samples has one more criterion. Each pixel map has a valid responsible range (R_i, R_{i+1}) where i is the map index. Two adjacent pixel maps share the same range bound. The range bound R_{i+1} is the upper bound of pixel map i and is the lower bound of pixel map $i + 1$. When the maximum of four regression targets lies within this range, the corresponding pixel is a positive sample. Algorithm 1 shows the positive sample matching process. The range numbers R_1, R_2, R_3, R_4, R_5, and R_6 for the original FCOS (FPNH-Ori) are listed in Table 2. They are hyperparameters for the FCOS detection algorithm.

3.2. Reallocating Computation Distribution. In determining the network configurations of the backbone, manually reducing the model size could be suboptimal, especially when the original configurations are based on an image classification dataset. SCRFD [6] points out that detecting small-scale faces requires more computation allocated in the shallow stage of the backbone. We transfer this design principle and apply it to VAN backbone design. In SCRFD, the backbone is based on ResNet, which has a hierarchical structure similar to VAN. A sequence of blocks is divided into four stages, and the deeper stage has a smaller spatial resolution. Design choices in the ResNet backbone are the number of blocks per stage and the output channel size per stage. We can find that these design choices have corresponding network configurations in the VAN backbone. Given the same face dataset, we believe that an optimal design choice in SCRFD can work well in other networks if they share similar structures. Therefore, we use the output channels and block numbers of SCRFD and define a new VAN backbone named VAN-Realloc. The configurations are shown in Table 1.

The computation reallocation happens not only within the backbone but also across network components. In the original full-sized FCOS, the FPN output channel is 256. When connecting FPN to the VAN-Reduce backbone, we use the same downscale factor 4, resulting in a 64-channel FPN output. For the VAN-Realloc backbone, the connected FPN has 24 output channels, which is consistent with the SCRFD design. Although the VAN-Realloc backbone has higher flops than the VAN-Reduce backbone, the gap is filled when complete structures are considered. FPN and FCOS head in VAN-Realloc detector induce fewer flops than in VAN-Reduce. Their computation distributions and performances are given in Section 4.2

3.3. Redesigning Positive Sample Matching. Although FCOS is anchor-free, the responsible range for each pixel map level plays a similar role as anchors in anchor-based detection algorithms, and they should be adjusted when facing a new dataset. During training, we resize the input image to 640 × 640. The bounding boxes are resized correspondingly, and most of them are below 64 × 64, as shown in Figure 5. If the matching criteria are not changed, the pixel map P8 is the most responsible for producing positive predictions. The other levels are less likely to get trained. This sample imbalance across different pixel maps downgrades the detection performance. Therefore, we modify the positive sample matching criteria and reduce pixel map numbers with corresponding feature pyramid levels. We name the

```
Input:
  $\mathcal{R}$ is a set of range numbers
  $\mathcal{G}$ is a set of bounding boxes in the input image
  $\mathcal{S}$ is a set of pixel maps' strides
Output:
  $P_k$ is the $k_{\text{th}}$ pixel map
for each level $k \in [1, \text{length}(\mathcal{S})]$ do
  build a mesh grid $M_i$ according to the stride $\mathcal{S}_k$ at $k_{\text{th}}$ level
  for each pixel $x_i, y_i \in M_i$ do
    calculate the pixel center coordinate $x_c, y_c$
    for each bounding box $g \in \mathcal{G}$ do
      if $x_c, y_c$ locates within $g$ then
        compute distances $d = (l, t, r, b)$ from the pixel center to the box's boundary
        if $\mathcal{R}_i < \max(l, t, r, b) < \mathcal{R}_{i+1}$ then
          if the pixel $x_i, y_i$ has not been marked positive then
            mark the pixel $x_i, y_i$ in $P_k$ a positive sample, assign regression target $d$, classification target $t$ and centerness score $c$
          else
            compare 4 distances $l, t, r, b$ with target $d$ and replace with $l, t, r, b$ if the corresponding value in $d$ is larger.
          end if
        end if
      end if
    end for
  end for
end for
```

Algorithm 1: Positive sample matching process.

Table 2: Detection structure configurations.

Detection config	FPN features	Pixel maps	Range numbers
FPNH-ori	N8, N16, N32, N64, N128	P8, P16, P32, P64, P128	0, 64, 128, 256, 512, inf
FPNH-rematch (ours)	N8, N16, N32	P8, P16, P32	0, 32, 64, inf
FPNH-quad (ours)	N8, N16, N32	P4, P8, P16	0, 16, 64, inf

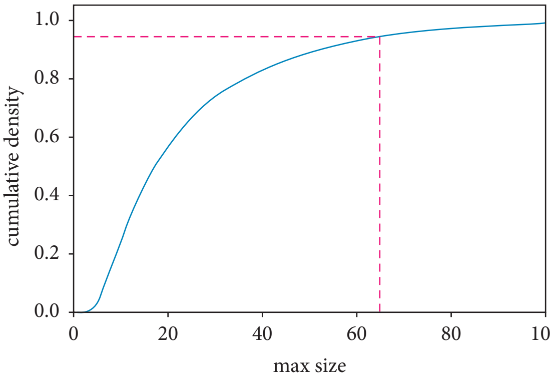


Figure 5: Cumulative density of the maximum size of bounding boxes in the WIDER FACE dataset.

modifications as FPNH-Rematch and show configurations in Table 2. Pixel maps P64 and P128 are removed. We put the maximum regressing targets below 32 in the pixel map P8. Around 72% of boxes can be assigned at this level. The targets between 32 and 64 are matched to the pixel map P16, and the remaining boxes are detected in the pixel map P32. The number reduction in the pixel map and feature pyramid is a better fit for the face dataset.

3.4. Quadruple Pixel Prediction. FCOS predicts only one bounding box at each pixel location, whereas the anchor-based SCRFD tiles two anchors per pixel, resulting in a doubled number of predictions. The WIDER FACE dataset is characterized by not only its small-scaled faces but also a considerable number of faces per image. Detectors that produce more predictions tend to perform better. Producing multiple predictions at the same pixel location for anchor-based detectors is easy due to their anchor-based nature. The positive sample matching rules are based on the Intersection over Union (IoU) between anchors and ground truth bounding boxes. For the anchor-free FCOS method, multiple predictions at the same location cause ambiguity in matching ground truths. To take the pros of multiple predictions and eliminate the ambiguity, we propose a quadruple pixel prediction method that defines a matching strategy. The idea is simple, and the implementation is straightforward. As shown in Figure 6, we quadruple the box predictions per location, which only requires the FCOS head to quadruple output channels. Then the four predictions at each pixel are reorganized and tiled as subpixel at the top-left, top-right, bottom-left, and bottom-right of the pixel, forming a new pixel map. If the original pixel map has a stride of S with four predictions per pixel, the new pixel map

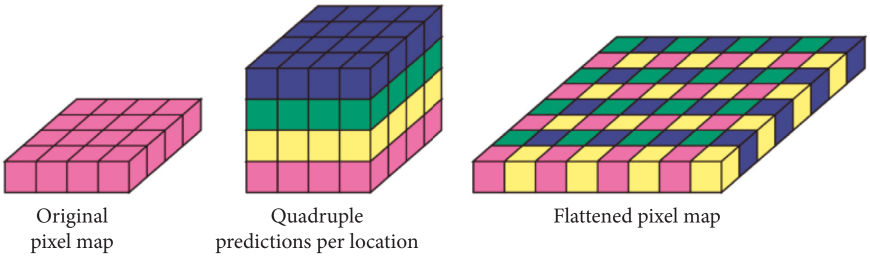


Figure 6: Quadruple pixel prediction.

can be equivalently viewed as a normal pixel map with one prediction per location but at stride $S/2$. In other words, we squeeze the pixel map on the channel level and expand it to the spatial level. Therefore, quadruple pixel predictions can be trained with no ambiguity. We present the configuration with quadruple pixel prediction in Table 2 and name it FPNH-Quad. As shown in the table, the new model produces the three neck outputs N8, N16, and N32, but the FCOS head turns them into P4, P8, and P16. No offset values are added to differentiate four predictions at the same location during training and inferring. We encourage the network to learn the offsets by itself because every quadrupled pixel is squeezed and expanded consistently. It is worth mentioning that although quadruple pixel prediction expands the pixel map by a factor of 2, other expansion values can be used to make more predictions.

4. Experiments and Analyses

4.1. Experimental Setup. We train and validate models on the WIDER FACE dataset [5]. The dataset contains 12880 images for training, 3226 for validation, and 16097 for testing. During training, we randomly crop and resize images to 640×640 without preserving the aspect ratio. Other data augmentation methods are used, such as random flip and random color jittering.

Moreover, we utilize a sample redistribute technique similar to SCRFD. Images are expanded at the ratio of 2 with a 50% chance at the beginning of preprocessing steps. To be specific, an image is pasted to a double-sized canvas. The pasted location is random, and the rest of the canvas is filled with a mean value of the WIDER FACE dataset. Since the random crop and resize operation is based on the original image size, a double-sized image leads to smaller bounding boxes when resized to the same 640×640. Therefore, more small faces are fed into the networks and encourage the networks to learn from them.

We train the networks for 300 epochs with a batch size of 16, and the training process uses the Adam optimizer. The learning rate has a linear warmup, increasing from $1e-6$ to $1e-3$ in 3 epochs. At 120 and 240 epochs, we decay the learning rate by 10. All models are trained from scratch, and no pre-trained weights are used to initialize parameters. We evaluate models on the validation set. During validation, we resize the image to 640×480 and use no test-time augmentation. The evaluation metric is AP at 0.5 IoU threshold on the WIDER FACE hard subset.

4.2. Computation Reallocation. To test the effectiveness of applying the SCRFD design principle, we train and validate two model configurations, VAN-Reduce-FPNH-Rematch and VAN-Realloc-FPNH-Rematch. VAN-Reduce-FPNH-Rematch takes a quarter of VAN-Tiny channels as the backbone, and FPN is reduced correspondingly. VAN-Realloc-FPNH-Rematch uses the VAN-Realloc backbone, and the FPN output channel size is guided by SCRFD. Both networks produce three pixel maps and use no quadruple pixel predictions. We present computation distributions of two model configurations and the WIDER FACE validation results in Table 3. The comparison is obvious: while both have close values in total flops, the backbone in VAN-Realloc-FPNH-Rematch has more significant proportions than that in VAN-Reduce-FPNH-Rematch. The first two stages of VAN-Realloc-FPNH-Rematch take up more than half of the total computation costs. The superior performance of VAN-Realloc-FPNH-Rematch, which beats its counterpart by 4.7%, indicates the necessity of this reallocation for computation.

4.3. Modification in Detection Structure. We present the numbers of positive samples for each pixel map under different detection configurations in Table 4. The numbers are accumulated through one training epoch. Since the training sample generation includes randomness, the positive samples reported in Table 4 are average values across epochs. FPNH-Ori is the original design in FCOS. It can be seen that almost all positive samples lie in the P8 pixel map. This extreme imbalance limits the network performance. With positive sample rematching and FPN reduction, FPNH-Rematch has a more balanced distribution of positive samples. When quadruple pixel prediction is introduced, we observe more matched cases in FPNH-Quad. We evaluate different detection configurations using the same VAN-Realloc backbone, and the performance is given in Table 4. FPNH-Rematch outperforms FPNH-Ori by 0.7% percent due to the balance across pixel maps. FPNH-Quad achieves the best performance at 70.5% due to the most positive samples.

4.4. Comparison with State-of-the-Art Model. We compare our best model (VAN-Realloc-FPNH-Quad) with state-of-the-art efficient face detectors (SCRFD series [6] and BlazeFace [17]) in flops, the number of parameters, and the detection accuracy under VGA resolution. We also report

Table 3: Flops distributions of VAN-Reduce-FPNH-Rematch and VAN-Realloc-FPNH-Rematch and their average precisions.

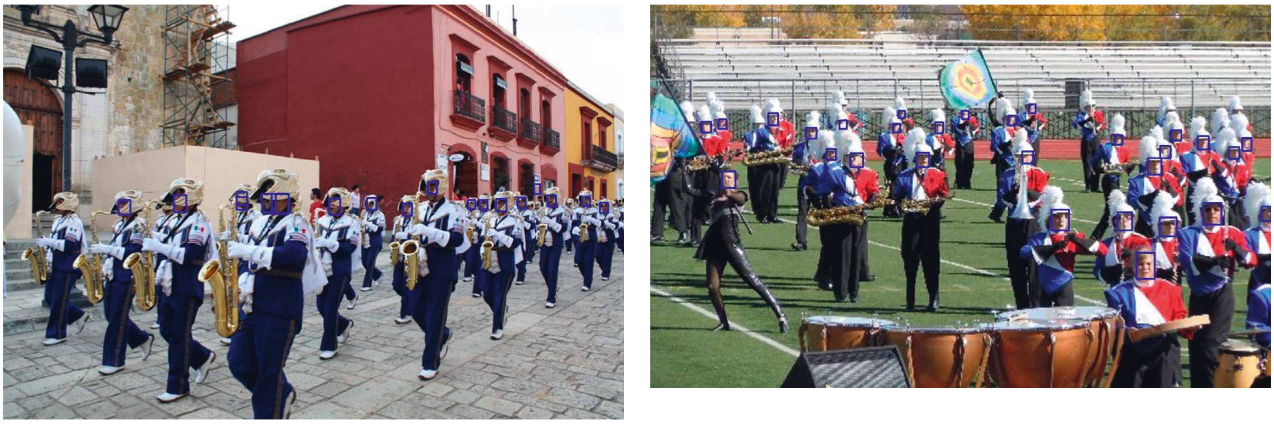
Model structure	VAN-Reduce-FPNH-Rematch		VAN-Realloc-FPNH-Rematch	
	Flops (G)	Percentage (%)	Flops (G)	Percentage (%)
Backbone stage 1	0.18	17.5	0.31	31.3
Backbone stage 2	0.04	3.9	0.34	34.3
Backbone stage 3	0.06	5.8	0.07	7.1
Backbone stage 4	0.02	1.9	0.03	3.0
FPN	0.24	23.3	0.04	4.0
FCOS head	0.49	47.6	0.20	20.2
Total	1.03	100	0.99	100
AP (%)	59.3		64.0	

Table 4: Positive samples matched per pixel map across different detection structures and average precisions under different configurations.

Pixel map stride	FPNH-Ori (K)	FPNH-Rematch	FPNH-Quad (K)
4	—	—	327.9
8	347.7	278.7	295.7
16	32.2	92.6	50.6
32	14.2	53.0	—
64	4.7	—	—
128	0	—	—
Total	398.8	424.3	674.2
AP (%)	63.4	64.0	70.5

Table 5: Comparison between VAN-Realloc-FPNH-Quad and other algorithms.

Algorithm	Flops (G)	Params (M)	AP (%)
SCRFD-2.5 G	2.53	0.67	77.9
SCRFD-0.5 G	0.51	0.57	68.5
BlazeFace	0.71	0.12	59.5
ResNet-Redesign-FPNH-Quad (ours)	1.03	0.37	69.5
VAN-Realloc-FPNH-Quad (ours)	1.05	0.30	70.5


(a)

(b)

Figure 7: Example results of the proposed method on the WIDER FACE dataset.

the FPNH-Quad structure with a ResNet backbone named ResNet-Redesign-FPNH-Quad. Its backbone is redesigned to achieve the same amount of computation as VAN-Realloc-FPNH-Quad. Results are shown in Table 5. Since BlazeFace does not report its performance on WIDER FACE, we train BlazeFace by ourselves. In Table 5, although SCRFD-2.5 G has the best AP of 77.9%, the cost is the largest model size and the most flops. SCRFD-0.5 G needs the least computation with a lower AP of 68.5%. The trained BlazeFace has the lowest AP but with the least parameters. VAN-Realloc-FPNH-Quad outperforms its ResNet-based counterpart by 1.0%, thanks to its attention mechanisms.

Our proposed model that ranks the second-best at AP of 70.5% needs only 1.05 Gflops, which is comparable to state-of-the-art models.

4.5. Engineering Applications. We show the detection results of our best model (VAN-Realloc-FPNH-Quad) in Figure 7 and suggest some potential engineering applications. Bounding box predictions are marked in blue. Figure 7(a) is an example of tiny faces with heavy occlusion. The detector is able to find the most faces, even with helmets. However, a few false predictions that detect a face twice are also produced by the model. In Figure 7(b), where partial illumination occurs, our model assigns a correct bounding box for almost every face. With the Internet of Things and big data [45, 46], face detection can find its application in V2X [47] or security surveillance systems [48].

5. Conclusion

We propose an efficient anchor-free detector that works at a low compute regime. The design absorbs the advancement in generic object detection and pays extra effort into tackling the tiny face problem. Using FCOS avoids the anchor-related hyperparameters. The visual attention backbone enhances the feature extraction by utilizing the LKA module. The design principle allocating more computation in shallow stages of the backbone improves detection performance, which is generalized from the ResNet-based networks to VAN-based networks. Sufficient and balanced positive samples in pixel maps facilitate detection performance, achieved by positive sample rematching and quadruple pixel prediction. With the techniques above, our efficient anchor-free detector arrives at 70.5% in AP with only 1.05 Gflops. While achieving competitive results with the state-of-the-art methods, we believe there is still room for improvement. Our detector takes the knowledge gained from SCRFD, which may limit the performance. A future direction is searching the detector's design space and finding design principles and network configurations for a more optimal design.

Data Availability

The dataset used to support this study is introduced by 10.1109/CVPR.2016.596 and is available at http://shuoyang1213.me/WIDERFACE/.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] G. G. Chrysos, E. Antonakos, S. Zafeiriou, and P. Snape, "Offline deformable face tracking in arbitrary videos," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 1–9, Santiago, Chile, 07-13 December 2015.

[2] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, 2016.

[3] F. Wang, X. Xiang, J. Cheng, and A. Loddon Yuille, "Normface: L2 hypersphere embedding for face verification," *Proceedings of the 25th ACM International Conference on Multimedia*, vol. 1, pp. 1041–1049, 2017.

[4] Y. Zhu, H. Cai, S. Zhang, C. Wang, and Y. Xiong, "Tinaface: strong but simple baseline for face detection," p. 13183, 2020, https://arxiv.org/abs/2011.13183.

[5] S. Yang, P. Luo, C. C. Loy, and X. Tang, "Wider face: a face detection benchmark," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5525–5533, Las Vegas, NV, USA, 27-30 June 2016.

[6] J. Guo, J. Deng, A. Lattas, and S. Zafeiriou, "Sample and computation redistribution for efficient face detection," 2021, https://arxiv.org/abs/2105.04714.

[7] J. Deng, J. Guo, E. Ververas, I. Kotsia, and S. Zafeiriou, "Retinaface: single-shot multi-level face localisation in the wild," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5203–5212, Seattle, WA, USA, 13-19 June 2020.

[8] Y. Liu, Xu Tang, J. Han, J. Liu, D. Rui, and X. Wu, "Hambox: delving into mining high-quality anchors on face detection," in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13043–13051, IEEE, Seattle, WA, USA, 13-19 June 2020.

[9] Z. Tian, C. Shen, H. Chen, and He Tong, "FCOS: Fully convolutional one-stage object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9627–9636, Seoul, Korea (South), October 2019 - 02 November 2019.

[10] M.-H. Guo, C.-Z. Lu, Z.-N. Liu, M.-M. Cheng, and S.-M. Hu, "Visual attention network," 2022, https://arxiv.org/abs/2202.09741.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, Las Vegas, NV, USA, 27-30 June 2016.

[12] P. Viola and M. Jones, "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*, vol. 1, IEEE, Kauai, HI, USA, 08-14 December 2001.

[13] T. Mita, T. Kaneko, and O. Hori, "Joint haar-like features for face detection," in *Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05)*, 17-21 October 2005.

[14] Y. Liu, F. Wang, J. Deng, Z. Zhou, B. Sun, and H. Li, "MogFace: towards a deeper appreciation on face detection," 2021, https://arxiv.org/abs/2103.11139.

[15] S. Zhang and C. Z. S. Z. Chi, "Refineface: refinement neural network for high performance face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 11, pp. 4008–4020, 2021.

[16] D. Xu and L. Y. Q. M. J. L. Wu, "OS-LFFD: a light and fast face detector with Ommateum structure," *Multimedia Tools and Applications*, vol. 80, no. 26-27, pp. 34153–34172, 2021.

[17] V. Bazarevsky, Y. Kartynnik, A. Vakunov, K. Raveendran, and M. Grundmann, "Blazeface: sub-millisecond neural face

detection on mobile gpus," 2019, https://arxiv.org/abs/1907.05047.

[18] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *Advances in Neural Information Processing Systems*, vol. 28, 2015.

[19] W. Liu, D. Anguelov, D. Erhan et al., "Ssd: single shot multibox detector," in *Proceedings of the ECCV 2016 European Conference on Computer Vision*, pp. 21–37, Netherlands, October 11–14, 2016.

[20] J. Redmon and F. Ali, "YOLO9000: better, faster, stronger," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7263–7271, Honolulu, HI, USA, 21-26 July 2017.

[21] J. Redmon and F. Ali, "Yolov3: an incremental improvement," 2018, https://arxiv.org/abs/1804.02767.

[22] H. Law and D. Jia, "Cornernet: detecting objects as paired keypoints," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 734–750, Munich, Germany, September 8–14, 2018.

[23] K. Duan, B. Song, L. Xie, H. Qi, Q. Huang, and T. Qi, "Centernet: keypoint triplets for object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6569–6578, Seoul, Korea (South), 27 October 2019 - 02 November 2019.

[24] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," p. 07850, 2019, https://arxiv.org/abs/1904.07850.

[25] T. Wolf, L. Debut, V. Sanh et al., "Transformers: state-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Melbourne, Australia, Octomber 2020.

[26] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[27] A. Dosovitskiy, L. Beyer, K. Alexander et al., "An image is worth 16x16 words: transformers for image recognition at scale," 2020, https://arxiv.org/abs/2010.11929.

[28] Ze Liu, Y. Lin, Y. Cao et al., "Swin transformer: hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022, Montreal, QC, Canada, 10-17 October 2021.

[29] W. Wang, E. Xie, Li Xiang et al., "Pyramid vision transformer: a versatile backbone for dense prediction without convolutions," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 568–578, Montreal, QC, Canada, 10-17 October 2021.

[30] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Proceedings of the ECCV 2020 European Conference on Computer Vision*, pp. 213–229, Glasgow, UK, August 23–28, 2020.

[31] Y. Wang, Z. Xu, X. Wang et al., "End-to-end video instance segmentation with transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8741–8750, Nashville, TN, USA, 20-25 June 2021.

[32] A. Srinivas, T.-Yi Lin, N. Parmar, J. Shlens, P. Abbeel, and A. Vaswani, "Bottleneck transformers for visual recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16519–16529, Nashville, TN, USA, 20-25 June 2021.

[33] M.-H. Guo, T.-X. Xu, J.-J. Liu et al., "Attention mechanisms in computer vision: a survey," 2021, https://arxiv.org/abs/2111.07624.

[34] J. Hu, Li Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 18-23 June 2018.

[35] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 18-23 June 2018.

[36] S. Xu, Y. Cheng, K. Gu, Y. Yang, S. Chang, and P. Zhou, "Jointly attentive spatial-temporal pooling networks for video-based person re-identification," in *Proceedings of the IEEE International Conference on Computer Vision*, Venice, Italy, 22-29 October 2017.

[37] X. Li, W. Wang, X. Hu, and J. Yan, "Selective kernel networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, 15-20 June 2019.

[38] A. G. Howard, M. Zhu, B. Chen et al., "Mobilenets: efficient convolutional neural networks for mobile vision applications," 2017, https://arxiv.org/abs/1704.04861.

[39] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, Salt Lake City, UT, USA, 18-23 June 2018.

[40] A. Howard, M. Sandler, G. Chu et al., "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1314–1324, Seoul, Korea (South), 27 October 2019 - 02 November 2019.

[41] I. Radosavovic, R. Prateek Kosaraju, R. Girshick, K. He, and P. Dollár, "Designing network design spaces," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10428–10436, Seattle, WA, USA, 13-19 June 2020.

[42] T.-Yi Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2117–2125, Honolulu, HI, USA, 21-26 July 2017.

[43] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," *Proceedings of the IEEE international conference on computer vision*, vol. 42, 2017.

[44] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: a metric and a loss for bounding box regression," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, 15-20 June 2019.

[45] M. Wu, L. Tan, and N. Xiong, "A structure fidelity approach for big data collection in wireless sensor networks," *Sensors*, vol. 15, no. 1, pp. 248–273, 2014.

[46] S. Huang and A. Liu, "A novel baseline data based verifiable trust evaluation scheme for smart network systems," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 3, 2020.

[47] K. Gao, F. Han, P. Dong, R. Du, N. Xiong, and R. Du, "Connected vehicle as a mobile sensor for real time queue length at signalized intersections," *Sensors*, vol. 19, no. 9, p. 2059, 2019.

[48] P. Yang, N. Xiong, and J. Ren, "Data security and privacy protection for cloud storage: a survey," *IEEE Access*, vol. 8, pp. 131723–131740, 2020.