Research Highlights (Required)

- Solve object detection by a pixel-wise paradigm.
- Anchor-free detector can completely avoid all the hyper parameters related to the anchor box.
- BiFPN effectively solved the problems related to multi-scale features.
- Soft-Weighted re-weights the quality of detection results to make detector more stable.

Computer Vision and Image Understanding
journal homepage: www.elsevier.com

An Anchor-Free Object Detector based on Soften Optimized Bi-directional FPN

Tao Zhang[a], Bo Jin[b,**], Wenjing Jia[c]

[a]School of Artificial Intelligence and Computer Science, Jiangnan University, Wuxi 214122, China
[b]School of Artificial Intelligence and Computer Science, Jiangnan University, Wuxi 214122, China
[c]Global Big Data Technologies Centre, University of Technology Sydney, Ultimo, NSW 2007, Australia

ABSTRACT

We propose an anchor-free object detector that combines a weighted bi-directional Feature Pyramid Network (BiFPN) and Soft Anchor Point Detector to address the object detection problem in a pixel–wise paradigm. The current mainstream object detection methods are anchor-based, which require to set hyper parameters such as scale and aspect ratio. This requires strong prior knowledge and can be difficult to design. Therefore, we propose an anchor-free detector that completely avoids the complex calculations and all the hyper parameters related to the anchor box by eliminating the predefined set of anchor boxes in an anchor-free way. Anchor-free detectors are essentially dense prediction methods. Although the huge solution space can yield high recall, simple anchor-free methods tend to return too many false positives, which leads to the problem of semantic ambiguity caused by the high overlap of object centers. Therefore, we propose BiFPN to alleviate the impact of high overlap which also effectively addresses the problems related to multi-scale features. Moreover, in order to utilize the power of feature pyramid better, we tackle the issues with a novel training strategy that involves two soften optimization techniques, i.e., soft-weighted anchor points and soft-selected pyramid levels. This training strategy further re-weights the quality of the detection results to make our detection results more stable.

1. Introduction

As the cornerstone of image understanding and computer vision, object detection is to solve instance segmentation [1, 2, 3], object tracking [4, 5, 6, 7], pose estimation [8, 9, 10, 11], a basis for more complex higher level of image description [12, 13, 14], action recognition [15, 16, 17] and other visual tasks. Object detection has a wide range of applications in many areas of artificial intelligence and information technology, including robotic vision, consumer electronics, security, autonomous driving, human-computer interaction, content-based image retrieval, intelligent video surveillance, and augmented reality. Due to the development of deep convolutional neural networks and well-labeled datasets, the performance of object detectors has been greatly improved.

All current mainstream detectors, such as Faster R-CNN [18], R-FCN [19], Mask R-CNN [2], SSD [20], DSOD [21], DCN [22], YOLOv2 [23], YOLOv3 [24] and YOLOv4 [25], etc. are anchor based approaches, which rely on a set of predefined anchor boxes. Although the use of anchor boxes has achieved great success, it also brings some inevitable drawbacks: 1) Use of predefined anchor boxes requires additional hyper-parameters such as the size, aspect ratio and number of anchor boxes, which settings have an impact on the detection performance, so they all need to be adjusted carefully; 2) Since the size and aspect ratio of the predefined anchor box are fixed, the detector may have difficulty in detecting objects with large changes in shape, especially small objects. The predefined anchor boxes negatively affect the generalization of the detectors, because for different datasets, it may be necessary to set different predefined anchor boxes to achieve the best effect of the detector.

Therefore, the scale variation is an urgent problem for object detection. To achieve scale invariability, the state-of-the-art detectors construct feature pyramids or multi-level feature towers. Also, different levels of feature layers are used to predict objects of different scales. For example, although the shallow feature

**Corresponding author: Tel.: +86-136-3591-7987;
e-mail: 13635917987@163.com (Bo Jin)

layer with larger length and width has less semantic information, it contains more detailed information and is suitable for predicting smaller objects. On the contrary, the deep feature layer does not have so much detailed information, but it contains richer semantic information and is suitable for predicting larger objects. The design of feature pyramids integrated with anchor boxes has achieved good performance on object detection benchmarks. However, in the training process, instances of different sizes are allocated to different feature layers for prediction, and this allocation rule is also artificially prescribed. Each instance always matches the closest anchor box based on the Intersection-over-Union (IoU) ratio. Obviously this has limitations. Why don't we let the model choose the most suitable feature layer for different instances? Moreover, each instance is limited to a single level, which leads to suboptimal utilization of the power of the feature pyramid.

In order to address the above two drawbacks, we propose an anchor-free optimization strategy to soft-select pyramid levels. Anchor-free detectors do not have predefined hyper-parameters associated with the anchor boxes, and so do not need to search for the optimal values for these hyper-parameters. Also, there is no need to set different predefined anchor boxes for different datasets, so it has better generalization ability. Besides, we propose a meta-selection network which can be jointly trained with the detector to predict the weight of the feature layer of each instance to achieve pyramid-level soft selection. In addition, not all pixels in the feature layer corresponding to an instance contain useful semantic information, and the amount of information contained in each pixel is also different. Therefore, we adopt a soft-weighted anchor point strategy, where the contribution of a positive anchor point to the network loss is re-weighted based on the anchor point's distance to the object center and the soft feature selection weights.

At the back of backbone network, we use BiFPN to perform multi-scale feature fusion effectively and efficiently to build a feature pyramid. Then, we scale the ground-truth boxes to the size corresponding to the feature pyramid and intercept the feature pyramid fragment. These fragments are then resized to the input shape of the meta-selection network, and passed into the meta-selection network to obtain the weight of each feature layer corresponding to each ground-truth box, which we call 'meta-selection weight'. Then, we use soft-weighted anchor points to weight the pixels on the feature pyramid and multiply them with the corresponding meta-selection weight to get our final soft-weight. Finally, the soft-weight is used for the calculation of regression and classification loss is used to train our model.

2. Related Work

2.1. Anchor-based Detectors

Since the RPN [18] was proposed in the field of object detection, anchor-based methods have become the mainstream of object detection models, such as Faster R-CNN, Mask R-CNN, Cascade R-CNN [26], Dynamic R-CNN [27], SSD, RetinaNet [28], and YOLO [29, 23, 24, 25]. In anchor-based detectors, predefined dense anchor boxes are generated by using the anchor mechanism that pre-sets several bounding boxes with different scales and aspect ratios for each pixel of the feature map, so that the network can perform target classification and bounding box coordinate regression on this basis directly. The predefined dense anchor boxes can effectively improve the network target recall ability, which is very effective for detecting small objects. Moreover, the addition of the predefined anchor boxes makes the training of the model more stable.

However, the hyper-parameters that need to be set for predefined anchor boxes, such as scale and aspect ratio, requires strong prior knowledge and are difficult to design. Also, there are so many redundant boxes produced. After all, the number of objects in an image is limited. Setting a large number of anchor boxes based on each anchor will generate a large number of easy-samples, that is, a background box that does not contain the object at all. This will cause a serious imbalance of positive and negative samples, which is one of the reasons why it is difficult for one-stage algorithms to compete with two-stage algorithms.

2.2. Anchor-free Detectors

Since the CornerNet [30] was proposed in August 2018, anchor-free object detection models have emerged one after another, and have reached a blowout state recently, announcing the rise of anchor-free object detection models. In fact, anchor-free is not a new concept. YOLOv1 [29] and DenseBox [31] are the earliest anchor-free models in the object detection field, and the recent ones such as FASF [32], FCOS [33], and FoveaBox [34] can all see the shadow of DenseBox. The anchor-free detectors can be roughly divided into anchor-point detection and key-point detection. The anchor-point detectors, such as Densebox, Unitbox [35], FCOS, FSAF or Foveabox, encode the ground-truth boxes as anchor points with corresponding point-to-boundary distances, where anchor points are pixels on the feature pyramid maps and their positions are associated with features. Key-point detectors, such as CornerNet, ExtremeNet [36], CenterNet [37], decode the key points into prediction boxes by predicting the positions of several key points of the bounding box, e.g., corners, centers, or extreme points.

2.3. Multi-scale Feature Fusion

Early target detection algorithms, whether single-stage or multi-stage, usually connect the detection head directly to the last layer of backbone's feature map for object detection. The output feature map resolution is 1/32 of the input image resolution, which is too small to be effective for object detection. Therefore, the MaxPooling of the last stage is generally removed or the conv layer with $stride = 2$ is changed to a conv layer with $stride = 1$ to increase the resolution of the last layer of feature maps. We call several layers with the same feature map resolution in the backbone as a stage. Later research found that in a single-stage object detection algorithm, a single-stage feature map cannot be used to effectively characterize objects of various scales at the same time. Therefore, the later object detection algorithms gradually developed to use the feature maps of different stages to form a feature pyramid network (FPN [38]) to characterize objects of different scales

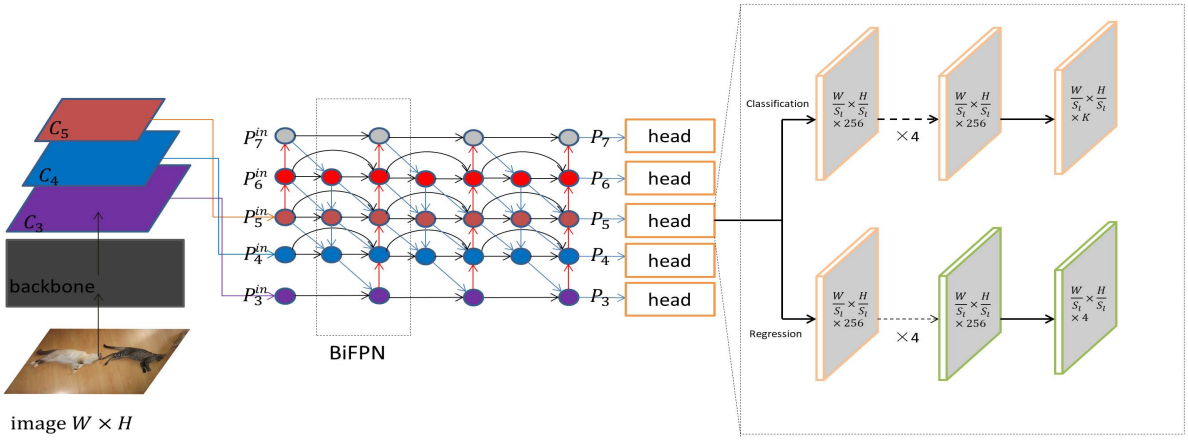

Fig. 1. Network architecture of the proposed anchor-free object detector. The backbone network is EfficientNet B0, and the feature pyramid network is BiFPN which is repeated three times and shared classification/regression network.

and then to perform object detection based on the feature pyramid, i.e., entered the FPN era. The evolution of FPN is mainly divided into four stages: 1) No fusion, such as SSD; 2) Top-down one-way fusion, such as Faster RCNN, Mask RCNN, Yolov3, RetinaNet, Cascade RCNN; 3) Simple bi-directional fusion, such as PANet [39]; 4) Complex bi-directional fusion, such as ASFF [40], NAS-FPN [41]. In this paper, iorder to obtain more excellent feature maps, we propose BiFPN based on bi-directional fusion.

3. Anchor-Free Object Detector

In this section, we mainly demonstrate our model from the following aspects: 1) An overall introduction to our network structure; 2) How to construct BiFPN to realize multi-scale feature fusion; 3) The design of supervision objective; 4) Anchor-point weighting strategy; 5) Pyramid-level weighting strategy; 6) The design of the loss function.

3.1. Network Architecture

The architecture of the anchor-free object detector is shown in Fig. 1. First, the picture of size $W \times H$ is input into the EfficientNet-B0 [42] backbone network, and the three feature layers C_i with the width and height of $W/2^i \times H/2^i$ with $i = 3$, 4, 5 can be obtained. Then, we simply process C_i to get P_l^{in} as the input of the BiFPN module to construct our feature pyramid with levels from P_3 through P_7. The resolution of P_l is $W/2^l \times H/2^l$, and l represents the level of feature layer. Specifically, a 1×1 conv layer is attached to the feature map C_3, C_4, and C_5 followed by the BatchNormalization [43]. It adjusts the number of channels to get P_3^{in}, P_4^{in}, and P_5^{in}. At the same time, a 1×1 conv layer is attached to the feature map C_5 followed by a MaxPooling2D layer with $pool\_size = 3$ and $strides = 2$. In addition to adjusting the number of channels, it also performs a downsampling operation to get P_6^{in}. Then we perform the same downsampling operation on P_6^{in} to get P_7^{in}. After obtaining the initial P_l^{in}, we can perform feature fusion through the BiFPN module to obtain the final feature pyramid with richer semantics and details. The resolution of each layer of the feature pyramid is different, so different feature layers will be used to detect objects of different scales. But they all share the same detection head. The detection head is divided into classification subnet and regression subnet. The classification subnet is used to predict the confidence of the spatial position of K objects for each anchor-point, where K represents the number of categories of objects in the dataset used for training. The regression subnet is used to predict the 4-dimensional class-agnostic offset for each anchor-point which represents the distance from the anchor point to the left, top, right, and bottom of the prediction box if the anchor-point's confidence exceeds the threshold.

3.2. BiFPN

The features generated by backbone are generally divided into stages, which are respectively denoted as C_1, C_2, C_3, C_4, C_5, C_6, C_7, etc. The number in it is the same as the number of the stage, which represents the number of times the resolution is halved. For example, C_2 represents the feature map output of Stage 2, the resolution is 1/4 of that of the input image, and C_5 represents the feature map output of Stage 5, which resolution is 1/32 of the input image. FPN [38] will take the features of different resolutions generated by the backbone as input, then perform feature fusion and output the fused features. The output features are generally marked with P as the number. For example, the input of FPN is C_l, and the output after fusion is P_l, where l represents the level of feature layers. This process can be expressed in mathematical formulas as $P_l = f(C_l)$. Before the FPN was proposed, the typical representative model of non-fusion, but also using multi-scale features, is the famous

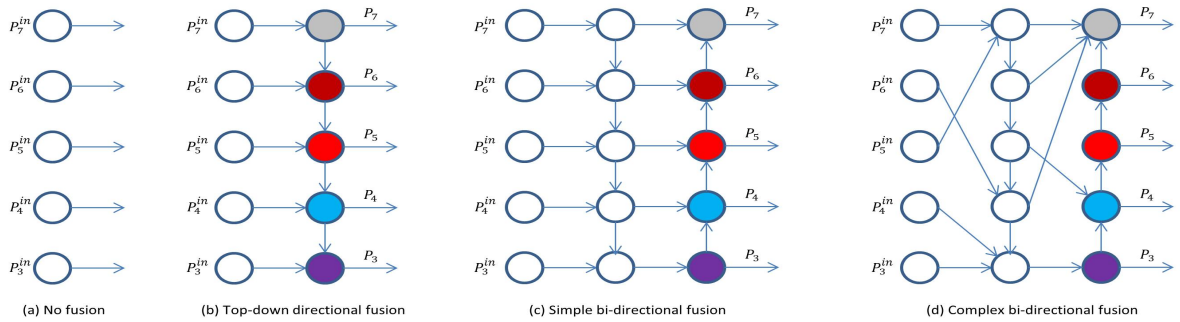

Fig. 2. The evolution of FPN-(a) No fusion; (b) Top-down direction fusion; (c) Simple bi-direction fusion; (d) Complex bi-direction fusion.

SSD, which directly uses the feature maps of different stages to be responsible for the detection of objects of different scales, as shown in Fig. 2(a). With the proposal of FPN, later algorithms such as Faster RCNN, Mask RCNN, Yolov3, RetinaNet and Cascade RCNN have introduced FPN. Although the details may be different, they are all top-down one-way fusions, as shown in Fig. 2(b). After that, PANet first proposed a bottom-up secondary fusion model, and added a bottom-up fusion path based on the FPN in Faster/Master/Cascade RCNN, as shown in Fig. 2(c). The proposal of PANet proves the effectiveness of two-way fusion. Because its two-way fusion is relatively simple, many papers have gone further in the direction of FPN and tried more complicated two-way fusion, as shown in Fig. 2(d).

Therefore, we also propose a complex two-way fusion feature network which is called bi-directional feature pyramid network (BiFPN). The specific details of its structure are shown in Fig. 3. When fusing features with different resolutions, we first resize them to the same resolution, and then aggregate them. Since different input features have different resolutions, they usually contribute unevenly to the output features. Therefore, we add an additional weight to each input and let the network learn the importance of each input feature. The weighting method is:

$$O = \sum_i \frac{w_i}{\epsilon + \sum_j w_j} \cdot I_i, \tag{1}$$

where I_i is the input, w_i is a learnable weight and Relu [44] is used for each update to ensure that its value is greater than 0, and ϵ is a small fractional, which main function is to prevent the value of the denominator is equal to 0. To facilitate understanding, we describe two fusion features at level 5 in BiFPN:

$$\begin{aligned} P_5^{td} &= C\left(\frac{w_1 \cdot P_5^{in} + w_2 \cdot R(P_6^{in}) + w_3 \cdot R(P_6^{td})}{w_1 + w_2 + w_3 + \epsilon}\right) \\ P_5^{out} &= C\left(\frac{(w_1^{'} \cdot P_5^{in} + w_2^{'} \cdot P_5^{td} + w_3^{'} \cdot R(P_6^{td}) + w_4^{'} \cdot R(P_4^{out}))}{(w_1^{'} + w_2^{'} + w_3^{'} + w_4^{'} + \epsilon)}\right) \end{aligned} \tag{2}$$

where C represents the depthwise separable convolutional [45] operation, and R represents the up-sampling or down-sampling operation for resolution matching.

3.3. Supervision Targets

Before explaining supervision targets, we must introduce the definition of some related concepts so that it can be easier to understand.

As we mentioned before, P_l represents the l-th layer in the feature layer, where l belongs to 3 to 5, and each feature layer P_l has $W/2^l \times H/2^l$ pixels, so we define the pixel points on the feature layer as anchor-points. For each anchor-point on each feature layer, we use p_{lij} to represent, where (i, j) with $i = 0, 1, ..., W/2^l - 1$ and $j = 0, 1, ..., H/2^l - 1$ representing the position of the pixel in the l-level feature layer. And each p_{lij} can find its corresponding position (X_{lij}, Y_{lij}) in the input image through the mapping relationship between $X_{lij} = 2^l(i + 0.5)$ and $Y_{lij} = 2^l(j + 0.5)$. For a ground-truth box, most of its semantic information is concentrated in the middle area, while the edge part contains less useful information. In addition, the information contained in the pixels on the edge that do not include instance is not helpful for the classification and regression prediction of the instance. On the contrary, they will provide wrong information and affect the accuracy of the prediction. Therefore, we define the effective box B_e as proportional regions of a ground-truth instance box $B = (c, x, y, w, h)$ controlled by constant scale factors ε, i.e., $B_e = (c, x, y, \varepsilon w, \varepsilon h)$ where c is the class ID, (x, y) is the box center and w, h are box width and height respectively. We set $\varepsilon = 0.2$. For each anchor-point, if and only if it corresponds to the position (X_{lij}, Y_{lij}) in the input image is within the range of the effective box B_e. We regard this anchor-point as a positive sample, otherwise it is a negative sample.

Supervision targets are composed of classification targets and regression targets. The classification target is a K-dimensional vector which number is equal to the number of all anchor-points in the five feature layers, and each dimension corresponds to one class. Therefore, for each effective anchor-point, its classification target is a K-dimensional vector with a value of 1 in the c-th dimension and a value of 0 in the remaining dimensions, and for a negative sample, its value is all 0. The regression target is a 4-dimensional vector agnostic to classes, which corresponds to the normalized distance $d = (d_l, d_t, d_r, d_b)$ from the anchor-point to the left, top, right, and bottom boundaries of B respectively. The calculation formula of distance is shown in

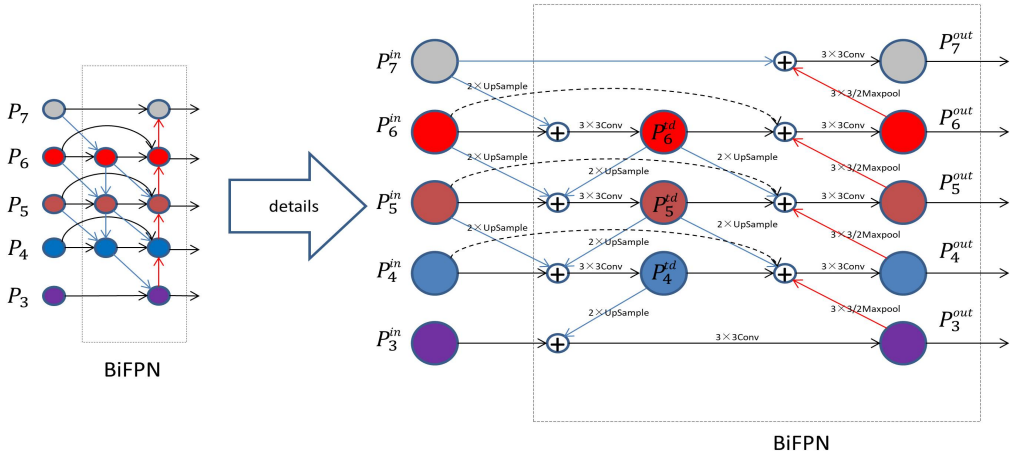

Fig. 3. The architecture of the bi-directional feature pyramid network (BiFPN). The Conv represents the depthwise separable convolution. Upsample and Maxpool represent the operation of doubling and halving the resolution, respectively.

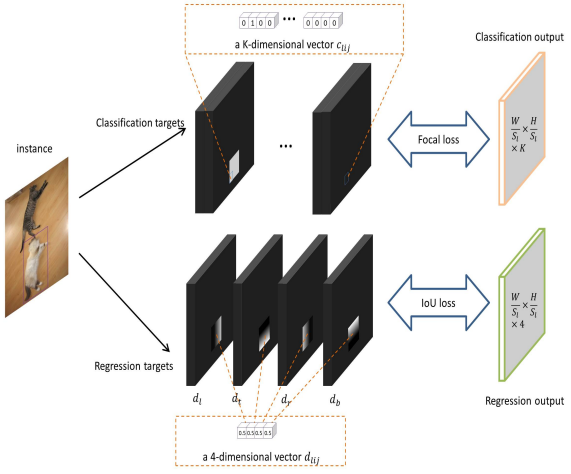

Fig. 4. Supervision targets for an instance. We only visualize the supervised targets of all anchor-points on a feature layer processed by the detection head.

Eq. (3),

$$\begin{aligned} d_l &= \frac{1}{S2^l}[X_{lij} - (x - w/2)] \\ d_t &= \frac{1}{S2^l}[Y_{lij} - (y - h/2)] \\ d_r &= \frac{1}{S2^l}[(x + w/2) - X_{lij}] \\ d_b &= \frac{1}{S2^l}[(y + h/2) - Y_{lij}] \end{aligned} \tag{3}$$

where S is the normalization scalar and is set as $S = 4.0$. Therefore, for each effective anchor-point, its regression target is a 4-dimensional vector with a value of d_l, d_t, d_r and d_b. For a negative sample we set its value to null. Finally, for each anchor-point p_{lij}, we have a corresponding classification target c_{lij} and positioning target d_{lij}. In order to facilitate understanding, we arrange the anchor-points on the same feature level into a matrix format, as illustrated in Fig. 4.

3.4. Soft-Weighted Anchor-Points

We observed that the anchor-point farther away from the center of the instance contains less semantic information but more background information, resulting in many low-quality prediction bounding boxes. In order to address this problem, we propose a simple and effective weighting strategy, adding different weights w_{lij} to each anchor-point p_{lij}. Our weighting strategy is to weight according to the distance between the current anchor-point and the anchor-point corresponding to the center position of the instance. For positive samples, the longer the distance, the smaller the weight so we can make the generation of the prediction box dependent more on the anchor-points near the center of the instance. For negative samples, since it is not used to generate prediction boxes, we keep their weights unchanged at 1. The specific formula is shown in Eq. (4) as:

$$w_{lij} = \begin{cases} \sqrt{\frac{min(d^l_{lij}, d^r_{lij}) min(d^t_{lij}, d^b_{lij})}{max(d^l_{lij}, d^r_{lij}) max(d^t_{lij}, d^b_{lij})}}, & p_{lij} \in p^+ \\ 1, & p_{lij} \in p^- \end{cases} \tag{4}$$

where d^l_{lij}, d^r_{lij}, d^t_{lij} and d^b_{lij} represent the normalized distance from p_{lij} to the left, right, top and bottom boundaries of the instance, respectively.

3.5. Soft-Selected Pyramid Levels

The anchor-based detector determines the feature level for the instance according to the IoU of the instance and all predefined boxes on each feature layer. For each instance, it only uses a certain layer in the feature pyramid. Our model uses anchor-free methods, so there is no such constraint and we can assign the same instance to multiple feature levels to make fuller use

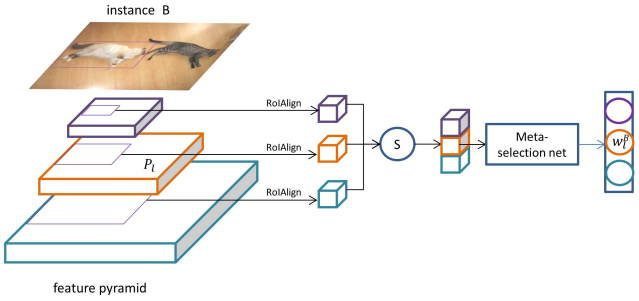

Fig. 5. The overall process of the weight prediction for soft-selected pyramid levels. For a more intuitive understanding, only three of the five layers of the feature pyramid are shown. "S" indicates the stack operation. w_l^B represents the weight of l-th level in the feature pyramid corresponding to instance B.

of the feature pyramid. FoveaBox [34] has demonstrated that assigning the same instance to multiple feature levels can improve the performance of the detector to a certain extent, and feature layers of different levels should make different contributions to the detection of the same instance. Therefore, we propose a method to allocate instances to each feature level in a certain proportion. This can also be understood as we allow each level in the feature pyramid to detect the same instance, but for the results of each level of detection, we have to apply different weights to achieve the effect that different levels of feature layers have made different contributions. We call this weight 'soft-selection weight'.

In order to obtain the optimal soft-selection weight, we propose a meta-selection network that can be trained with the entire detection model, and use this network to generate soft-selection weight. The detailed architecture of the meta-selection network is shown in Table 1.

Table 1. Architecture of the meta-selection network.

Layer Type	Output Size	Layer Set	Batch Normalization	Activation
Input	$1280 \times 7 \times 7$	n/a	N	n/a
conv	$256 \times 5 \times 5$	3×3,256	Y	ReLU
conv	$256 \times 3 \times 3$	3×3,256	Y	ReLU
conv	$256 \times 1 \times 1$	3×3,256	Y	ReLU
fc	5	n/a	N	Softmax

It consists of three 3×3 conv layers and one fully-connected layer. Each conv layer with no padding is followed by Batch-Normalization and ReLU activation functions, and the activation function of fully-connected layer is softmax. We use cross entropy loss to optimize it and for each instance its corresponding ground-truth is a one-hot vector whose value is equal to the level of the optimal feature pyramid corresponding to the instance selected by the FSAF module.

Our overall process is shown in Fig. 5. First, we extract the instance-dependent feature responses from all the pyramid levels. Then we use RoIAlign [2] layer to resize the width and height of the feature responses to 7×7 and stack them in the dimension of the channel. Finally, the stacked features are input into the meta-selection network to obtain a 5-dimensional vector, which is the soft-selection weight. Thus, we associate the instance B with the weight w_l^B of each level in the feature pyramid. On this basis, if the anchor-point p_{lij} of the instance-dependent feature responses is inside B_e, then we can combine the two weights to get the final weight of the anchor-point, as shown in Eq. (5).

$$w_{lij} = \begin{cases} w_l^B \sqrt{\frac{min(d_{lij}^l, d_{lij}^r) min(d_{lij}^t, d_{lij}^b)}{max(d_{lij}^l, d_{lij}^r) max(d_{lij}^t, d_{lij}^b)}}, & \exists B, p_{lij} \in B_e \\ 1, & otherwise \end{cases} \tag{5}$$

3.6. Loss Functions

For the classification subnet, we use focal loss (l_{FL}) [28] for training to overcome the imbalance between positive and negative anchor-points with hyperparameters $\alpha = 0.25$ and $\gamma = 2.0$. For the regression subnet, we use IoU loss (l_{IoU}) for training, and all negative anchor-points do not participate in the calculation of IoU loss [35]. For each anchor-point p_{lij}, the output of the classification subnet is a K-dimensional vector $\hat{c}_{lij}$, and the output of the regression subnet is a 4-dimensional vector $\hat{d}_{lij}$, and its loss (L_{lij}) calculation formula is shown in Eq. (6).

$$L_{lij} = \begin{cases} l_{FL}(\hat{c}_{lij}, c_{lij}) + l_{IoU}(\hat{d}_{lij}, d_{lij}), & p_{lij} \in p^+ \\ l_{FL}(\hat{c}_{lij}, c_{lij}), & p_{lij} \in p^- \end{cases} \tag{6}$$

where c_{lij} and d_{lij} are the targets of classification and regression respectively. So far, we have obtained the loss of each anchor-point. We then multiply it with the corresponding weight w_{lij} to get its weighted loss. Finally, the total loss of the entire model is the total weighted loss of anchor-points divided by the total weight of all positive anchor-points plus the cross entropy loss of the meta-selection network, as in Eq. (7).

$$L = \frac{1}{\sum_{p_{lij} \in p^+} w_{lij}} \sum_l \sum_{ij} w_{lij} L_{lij} + \lambda L_{meta-net} \tag{7}$$

where $L_{meta-net}$ is the cross entropy loss of the meta-selection network. Since the model is used for object detection, $L_{meta-net}$ cannot account for too much in the total loss, so λ is used to control its ratio, we set $\lambda = 0.1$.

4. Experiments

We conduct experiments on the PASCAL VOC dataset [46, 47] and MS COCO dataset [48]. For the PASCAL VOC dateset, the training data is PASCAL-VOC2007 *trainval* set and *test* set plus PASCAL-VOC2012 *trainval* set, which contains approximately 21.5k images. For the MS COCO dataset, the training data is *train2017* set, which contains approximately 118k images. All ablation experiments are performed on PASCAL-VOC2007 *val* (containing 2.5k images), PASCAL-VOC2012 *val* (containing 5k images) and MS COCO *val* set (containing 5k images). When comparing with other state-of-the-art detectors, we report our main research results on the PASCAL-VOC2012 *test* and MS COCO *test-dev*.

4.1. Inference Details

Anchor-free object detector is a single FCN [49] composed of EfficientNet B0 backbone, BiFPNs, a classification subnet and a regression subnet. Since the meta-selection network is only used to train the detection head, it does not participate in inference. As such, inference only involves forwarding the image through the network in the form of full convolution. Then, each anchor point p_{lij} generates classification prediction c_{lij} and localization prediction d_{lij} through the same classification subnet and regression subnet. The d_{lij} is decoded by the reverse Eq. (8) of Eq. (3) to obtain the bounding box.

$$\begin{aligned} x_1 &= X_{lij} - d_l \cdot S2^l + w/2 \\ y_1 &= Y_{lij} - d_t \cdot S2^l + h/2 \\ x_2 &= X_{lij} + d_r \cdot S2^l - w/2 \\ y_2 &= Y_{lij} + d_b \cdot S2^l - h/2 \end{aligned} \tag{8}$$

where (x_1, y_1) and (x_2, y_2) represent the coordinates of the upper left corner and the lower right corner of the predicted bounding box, respectively. Unless otherwise noted, we only decode the d_{lij} corresponding to at most 1k top-scoring anchor points in each pyramid level to the prediction bounding boxes, after thresholding detector confidence at 0.05. After that, the top prediction bounding boxes from each pyramid level are merged and non-maximum suppression is applied with a threshold of 0.5 to yield the final detections.

4.2. Training Details

The entire detection network and meta-selection network are jointly trained using Adam optimizer. The training process is divided into two steps in total. In the first step, we freeze the backbone for 50 epoch rough training with the initial learning rate being 10^{-3} and a batchsize of 32 images. In the second step, we unfreeze the backbone and freeze all BN layers for 50 epoch precision training with the initial learning rate being 10^{-5} and a batchsize of four images. The remaining parameters of the Adam optimizer are all default values, and each epoch is performed 1K iterations for a total of 100k iterations. We initialize the backbone network with the weights provided by EfficientNet B0. The initialization of BiFPN is the same as [50]. Initializing all layers with a Gaussian weight and bias $b = 0$ in the meta-selection network and subnets (except for the final layer), and for batch normalization the momentum and epsilon are set as 0.99 and 0.001. The final conversion layer of the classification subnet is initialized with bias $b = -\log((1-\pi)/\pi)$, where $\pi = 0.01$, and a Gaussian weight. The final conversion layer of the regression subnet is initialized with a bias $b = 0.1$, and also a Gaussian weight. All the Gaussian weights are filled with $\sigma = 0.01$.

4.3. Ablation Studies

In all ablation studies, the input image scale used for training and testing the model is 512×512 pixels. We evaluated the contribution of BiFPN, soft-weighted anchor points and soft-selected pyramid levels to our detector. But before that, we have to determine the constant scale factor ε that controls the size of the effective box B_e. We first apply different ε to the FSAF detector with EfficientNet-B0 backbone and BiFPN. The results are reported in Table 2. Obviously, setting ε to 0.2 is the best choice.

Table 2. Varying ε for the size of effective boxes B_e. Detector is FSAF with EfficientNet-B0 backbone and BiFPN for all experiments in this table. Val represents the validation dataset.

mAP / ε / Val	0.1	0.2	0.3	0.4	0.5
VOC12-val	76.8	**78.4**	77.3	76.7	76.1
VOC07-val	77.5	**79.2**	78.0	77.2	76.5
COCO-val	36.6	**37.6**	36.8	36.3	35.8

Table 3. The impact of different multi-scale feature fusion strategies on detection

validation dataset	feature fusion strategy	mAP
VOC12-val	No fusion	68.8
	FPN	73.6
	PANet	77.3
	BiFPN	**78.4**
VOC07-val	No fusion	69.1
	FPN	74.0
	PANet	77.8
	BiFPN	**79.2**
COCO-val	No fusion	32.2
	FPN	35.5
	PANet	36.8
	BiFPN	**37.6**

BiFPN can better fuse multi-scale features to build a better feature pyramid. First, we use different multi-scale feature fusion networks to train our model. To individually verify the importance of BiFPN, we did not apply soft-weighted anchor points and soft-selected pyramid levels methods, but used FSAF's Online Feature Selection [32] method to assign a feature level to each instance. Moreover, for the fairness of ablation research, we designed each multi-scale feature fusion network as a separate module like BiFPN and repeated it three times to connect it behind the backbone network. The architecture of No fusion, FPN, PANet and BiFPN are shown in Fig. 2(a), Fig. 2(b), Fig. 2(d) and Fig. 3, respectively.

The specific details of the ablation experiment are as follows: starting from the FSAF detector with EfficientNet-B0 backbone and No fusion, first we use top-down FPN instead of No fusion, which improves the accuracy by 4.8 mAP on the VOC12-val. Then, we replace FPN with PANet, which further improves the accuracy by about 3.7 mAP. Finally, we use BiFPN to replace PANet, which improves the accuracy by nearly 1.1mAP on the basis of the previous one. It is worth noting that the details of obtaining the input P_l^{in} of each multi-scale fusion network are mentioned in Section 3.1, and No fusion directly uses the P_l^{in} as

Table 4. Ablative experiments for SW and SS on VOC07-val, VOC12-val and COCO-val. SW represents the soft-weighted anchor points, SS represents the soft-selected pyramid levels and OFS represents Online Feature Selection method of FSAF, respectively.

Validation Dataset	OFS	SW	SS	mAP
VOC12-val	√			78.4
	√	√		80.5
			√	80.3
		√	√	**82.2**
VOC07-val	√			79.2
	√	√		81.3
			√	80.9
		√	√	**83.1**
COCO-val	√			37.6
	√	√		39.8
			√	39.1
		√	√	**42.5**

the output. Results are reported in Table 3. These results show that our proposed BiFPN can better fuse multi-scale features compared to other feature fusion networks, thereby improving the accuracy of the detection.

Soft-weighted anchor points and soft-selected pyramid levels improve the detection performance. In order to study whether the soft weighted anchor points can improve the performance of the detection head, we apply it to train the FSAF detector with EfficientNet-B0 backbone and BiFPN and the method of assigning feature levels to each instance is still the Online Feature Selection method of FSAF. Similarly, in order to study whether the soft-selected pyramid levels can improve the detection performance, we only use it to replace the Online Feature Selection method of FSAF to assign feature levels to each instance without using soft-weighted anchor points. Finally, we use both soft-weighted anchor points and soft-selected pyramid levels to train the detector to study whether the combined effect of the two can improve the detection performance. Results are reported in Table 4. It is evident that soft-weighted anchor points and soft-selected pyramid levels methods can significantly improve the detection performance. Taking the ablation study on VOC12-val as an example, the experimental results of using only either soft-weighted anchor points or soft-selected pyramid levels have increased by 2.1 mAP and 1.9 mAP respectively, and the combined effect of the two has increased by 3.8 mAP.

4.4. Comparison to State of the Arts

We evaluate our final detector on the PASCAL-VOC2012 *test* set and MS COCO *test-dev* set to compare with the state-of-the-art detectors. For all experiments, we have to preprocess the image. Specifically, the long side of the image is first scaled to 512 pixels, and then the short side is scaled with the same ratio and filled to 512 pixels with gray bars, where the gray bars are RGB three channels with a value of 128. Finally, divide the value of each pixel by 255 and subtract the mean value [0.485, 0.456, 0.406] and divide by the standard deviation [0.229, 0.224, 0.225] for normalization. Other training details are the same as in Section 4.2.

4.4.1. PASCAL VOC 2012

Following the protocol of VOC 2012, we submit the detection results of our method to the public testing server for evaluation. We use VOC 2012 trainval set for *training*, and test on VOC 2012 *test* set. Table 5 presents the comparison results. Compared with the state of the art one-stage anchor-free methods, our method achieves 0.5 mAP higher than SAPD [53]. Compared with the latest one-stage anchor-base methods, our method outperforms the best performing ASSD [52] by 0.4 mAP. Compared with the recent two-stage methods, our method achieves a 1.6 mAP improvement above the top-performing Cascade R-CNN. Finally, we show some detection results of our method in Fig. 6.

4.4.2. MS COCO

In addition to PASCAL VOC, we also evaluate our method on MS COCO. Following the protocol in MS COCO, we use the *trainval35k* set for training and evaluate the results from *test-dev* evaluation server. Except that a different backbone network is used, the rest of the training details are the same as in section 4.2. Table 6 shows the results on MS COCO *test-dev* set. We report the results of four series of backbone models. With ResNeXt-101-64x4d, our method outperforms the ATSS with the same backbone ResNeXt-101-64x4d by 0.7% in AP. With ResNet-101, our method also outperforms other two-stage detectors such as TridentNet with the same backbone by 1.4% in AP. With ResNeXt-101-64x4d-DCN as the backbone, our method achieves 47.6% in AP. Moreover, if we use the Copy-Paste [56] data augmentation method, our method can be further improved to 49.3% in AP. We show some qualitative results on the MS COCO *test-dev* in Fig.7.

Conclusion

This paper has proposed a model that combines the training strategy of soft-weighted anchor points and soft-selected pyramid levels with BiFPN, which not only builds a better feature pyramid but also solves the problem of attention bias and feature selection in anchor-free object detection. As shown in the experiment, our model is comparable to the anchor-based detectors of the YOLO and SSD series, and it can also completely avoid all calculations and hyperparameters related to the anchor boxes.

Acknowledgment

We would like to thank the authors of [32], [53] and [50] for providing feature selective anchor-free module, the soft anchor-point training strategy and the design idea of BiFPN. Finally, we would like to thank the providers of the PASCAL-VOC dataset and MS COCO dataset. This work is supported by the National Natural Science Foundation of China (NO. 61702226); the 111 Project (B12018); the Natural Science Foundation of Jiangsu Province (NO. BK20170200); Open

Table 5. Object detection results of our method with the EfficientNet-B0 backbone, BiFPN module, soft-weighted anchor points and soft-selected pyramid levels strategies vs. state-of-the-art one-stage methods, i.e., RetinaNet, SSD, DSSD, FSSD [51], ASSD [52], YOLOv2, YOLOv3, YOLOv4, FSAF, CenterNet, SAPD [53] and two-stage methods detectors, i.e., Faster R-CNN, OHEM [54], Cascade R-CNN, Soft-NMS [55] on PASCAL-VOC2012 *test*.

Method	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Two-stage methods:																					
Faster R-CNN	74.4	84.4	77.7	83.9	60.4	49.4	84.6	70.0	94.7	58.7	73.8	57.0	93.2	84.9	76.7	80.1	46.3	77.6	69.0	86.6	78.3
OHEM	75.9	87.1	80.6	85.6	61.0	52.8	83.0	70.7	96.0	60.6	74.7	54.3	94.0	88.4	82.8	82.3	49.2	78.8	66.9	89.6	79.8
Cascade R-CNN	80.5	89.5	83.6	87.2	68.4	56.0	85.2	75.8	97.5	63.7	82.5	66.9	95.2	89.3	86.9	85.5	57.0	84.8	79.4	92.2	84.1
Soft-NMS	80.3	89.1	82.4	88.2	66.7	57.9	86.1	74.1	97.7	67.5	83.4	64.9	94.7	90.3	85.3	85.4	57.4	81.5	77.3	91.9	84.4
One-stage methods:																					
RetinaNet	75.0	85.7	77.4	85.2	62.2	52.5	84.3	70.3	95.7	59.8	73.9	54.9	93.2	84.1	80.4	82.0	48.9	76.8	65.7	87.7	78.7
SSD	78.9	88.5	83.9	87.4	64.1	56.1	86.4	74.1	95.6	73.8	82.7	61.9	95.2	88.9	85.3	83.7	53.0	81.8	73.5	91.3	80.9
DSSD	79.7	89.7	81.8	87.3	66.5	56.1	87.1	74.8	97.7	65.4	78.5	61.7	95.7	87.7	86.6	84.9	64.8	83.6	78.0	92.4	83.1
FSSD	80.8	89.7	84.7	88.3	69.8	57.5	86.5	74.4	97.1	67.7	84.9	65.4	93.1	90.2	86.7	85.1	57.1	82.8	78.7	92.2	83.9
ASSD	81.7	90.3	85.0	88.9	69.5	58.1	**88.7**	**76.8**	97.3	69.6	**85.4**	65.7	96.1	90.1	87.2	85.5	57.9	83.8	80.3	92.2	**85.5**
YOLOv2	77.3	86.5	83.0	86.2	63.0	54.6	85.9	71.2	96.0	62.1	78.3	57.9	93.8	87.7	83.1	82.7	50.4	80.9	73.5	89.2	80.7
YOLOv3	78.3	89.2	82.1	86.3	65.3	56.1	85.4	69.9	94.7	65.9	78.5	63.5	94.7	88.4	83.9	84.8	52.8	74.6	76.7	92.2	81.8
YOLOv4	80.7	90.0	83.0	88.1	67.7	59.2	86.0	74.7	97.5	67.8	84.0	65.1	95.7	89.0	85.8	85.6	56.9	82.9	78.8	92.2	83.3
FSAF	77.7	88.3	80.0	84.0	62.9	54.7	85.5	72.4	96.4	60.2	79.7	62.9	94.4	87.9	81.6	83.3	51.8	81.8	74.9	90.2	81.5
CenterNet	79.7	89.9	84.3	88.6	68.3	52.9	85.3	74.3	97.2	65.7	80.5	65.5	95.1	89.6	86.8	84.6	53.4	82.9	74.8	92.4	81.3
SAPD	81.6	89.7	**86.1**	**89.1**	70.3	58.8	87.1	75.8	97.4	68.9	84.5	**67.8**	95.5	**90.6**	87.3	86.4	56.6	84.0	78.8	**93.4**	84.4
Ours	**82.1**	**90.5**	85.8	88.3	**71.2**	**59.4**	88.3	76.7	**98.0**	**69.2**	84.2	66.7	**96.3**	89.5	**88.3**	**86.7**	**58.2**	**85.1**	**82.4**	92.5	84.6

Fig. 6. Some test results on PASCAL-VOC2012 *test*. As shown in the figure, our method is suitable for various objects, including crowded, occluded, highly overlapping, extremely small and very large objects. And it has good performance for bottle and sheep, which are difficult to detect in the PASCAL-VOC dataset.

Table 6. Performance comparison with state-of-the-art one-stage methods and two-stage methods on MS COCO *test-dev*.

Method	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods:							
Faster R-CNN w FPN [38]	ResNet-101	36.2	59.1	39.0	18.2	39.0	48.2
Mask R-CNN [2]	ResNet-101	38.2	60.3	41.7	20.1	41.1	50.2
Cascade R-CNN [26]	ResNet-101	42.8	62.1	43.6	23.7	45.5	55.2
SNIPER [57]	ResNet-101	46.1	67.0	51.6	29.6	48.9	58.1
RepPoints [58]	ResNet-101	41.0	62.9	43.3	23.6	44.1	51.7
RepPoints [58]	ResNet-101-DCN	45.0	66.1	49.0	26.6	48.6	57.5
TridentNet [59]	ResNet-101	42.7	63.6	46.5	23.9	46.6	56.6
TridentNet [59]	ResNet-101-DCN	46.8	67.6	51.5	28.0	51.2	60.5
One-stage methods:							
RetinaNet [28]	ResNet-101	39.1	59.1	42.3	21.8	42.7	50.2
CornerNet [30]	Hourglass-104	40.6	56.4	43.2	19.1	42.8	54.3
CenterNet [37]	Hourglass-104	42.1	61.1	45.9	24.1	45.5	52.8
FSAF [32]	ResNet-101	40.9	61.5	44.0	24.0	44.2	51.3
FSAF [32]	ResNeXt-101-64x4d	42.9	63.8	46.3	26.6	46.2	52.7
FoveaBox [34]	ResNet-101	40.6	60.1	43.5	23.3	45.2	54.5
FoveaBox [34]	ResNeXt-101-64x4d	42.1	61.9	45.2	24.9	46.8	55.6
FCOS [33]	ResNet-101	41.5	60.7	45.0	24.4	44.8	51.6
FCOS [33]	ResNeXt-101-64x4d	44.7	64.1	48.4	27.6	47.5	55.6
FreeAnchor [60]	ResNet-101	43.1	62.2	46.4	24.5	46.1	54.8
FreeAnchor [60]	ResNeXt-101-64x4d	44.9	64.3	48.5	26.8	48.3	55.9
SAPD [53]	ResNet-101	43.5	63.6	46.5	24.9	46.8	54.6
SAPD [53]	ResNeXt-101-64x4d	45.4	65.6	48.9	27.3	48.7	56.8
ATSS [61]	ResNet-101	43.6	62.1	47.4	26.1	47.0	53.6
ATSS [61]	ResNeXt-101-64x4d	45.6	64.6	49.7	28.5	48.9	55.6
Ours	ResNet-101	44.1	63.6	47.8	25.6	47.5	55.2
Ours	ResNet-101-DCN	46.7	66.4	50.1	27.4	49.6	59.8
Ours	ResNeXt-101-64x4d	46.3	66.0	49.8	26.8	49.3	57.7
Ours	ResNeXt-101-64x4d-DCN	47.6	67.9	51.2	29.5	50.6	60.3
Ours + Copy-Paste	ResNeXt-101-64x4d-DCN	49.3	68.3	53.9	28.8	52.0	61.3

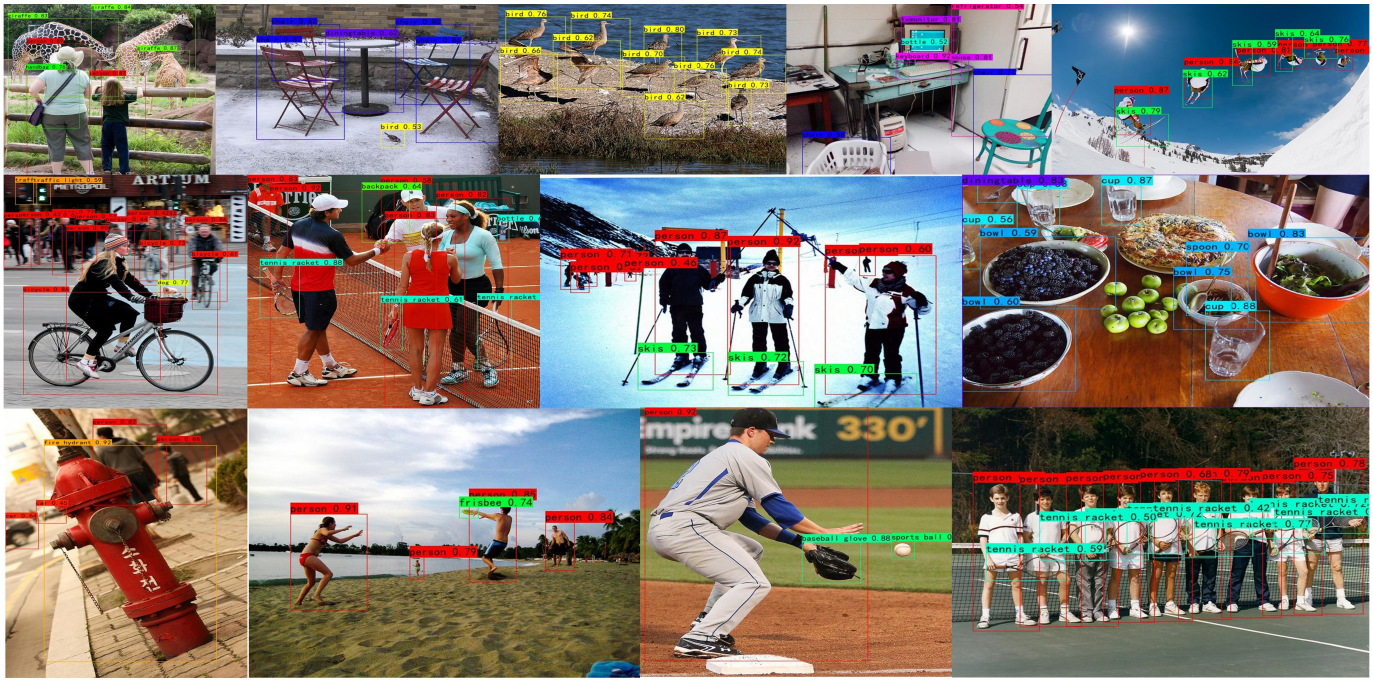

Fig. 7. Qualitative results of our method on the COCO *test-dev* set (corresponding to 44.1% mAP). ResNet-101 is used as the backbone network. The training data is COCO *trainval35k*.

Fund of Key Laboratory of Urban Land Resources Monitoring and Simulation, Ministry of Land and Resources (NO. KF-2018-03-065); the Fundamental Research Funds for the Central Universities (NO. JUSRP11854,NO. JUSRP11851).

References

[1] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A. L. Yuille, Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, IEEE transactions on pattern analysis and machine intelligence 40 (4) (2017) 834–848.

[2] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask r-cnn, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 2961–2969.

[3] K. Konyushkova, R. Sznitman, P. Fua, Geometry in active learning for binary and multi-class image segmentation, Computer vision and image understanding 182 (2019) 1–16.

[4] Y. Wu, J. Lim, M.-H. Yang, Online object tracking: A benchmark, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2013, pp. 2411–2418.

[5] M. Wang, Y. Liu, Z. Huang, Large margin object tracking with circulant feature maps, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4021–4029.

[6] M. Danelljan, G. Bhat, F. Shahbaz Khan, M. Felsberg, Eco: Efficient convolution operators for tracking, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 6638–6646.

[7] F. Wu, S. Peng, J. Zhou, Q. Liu, X. Xie, Object tracking via online multiple instance learning with reliable components, Computer Vision and Image Understanding 172 (2018) 25–36.

[8] Z. Cao, T. Simon, S.-E. Wei, Y. Sheikh, Realtime multi-person 2d pose estimation using part affinity fields, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 7291–7299.

[9] Y. Kawana, N. Ukita, J.-B. Huang, M.-H. Yang, Ensemble convolutional neural networks for pose estimation, Computer Vision and Image Understanding 169 (2018) 62–74.

[10] J. Huang, Z. Zhu, F. Guo, G. Huang, The devil is in the details: Delving into unbiased data processing for human pose estimation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 5700–5709.

[11] J. Y. Chang, K. M. Lee, 2d–3d pose consistency-based conditional random fields for 3d human pose estimation, Computer Vision and Image Understanding 169 (2018) 52–61.

[12] O. Vinyals, A. Toshev, S. Bengio, D. Erhan, Show and tell: A neural image caption generator, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 3156–3164.

[13] J. Lu, C. Xiong, D. Parikh, R. Socher, Knowing when to look: Adaptive attention via a visual sentinel for image captioning, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 375–383.

[14] X. Yang, K. Tang, H. Zhang, J. Cai, Auto-encoding scene graphs for image captioning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 10685–10694.

[15] H.-H. Pham, L. Khoudour, A. Crouzil, P. Zegers, S. A. Velastin, Exploiting deep residual networks for human action recognition from skeletal data, Computer Vision and Image Understanding 170 (2018) 51–66.

[16] J. Carreira, A. Zisserman, Quo vadis, action recognition? a new model and the kinetics dataset, in: proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 6299–6308.

[17] Y. Quan, Y. Chen, R. Xu, H. Ji, Attention with structure regularization for action recognition, Computer Vision and Image Understanding 187 (2019) 102794.

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

[19] K. Dai, Y. R-FCN, Object detection via region-based fully convolutional networks. arxiv preprint, in: arXiv preprint, 2016.

[20] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A. C. Berg, Ssd: Single shot multibox detector, in: European conference on computer vision, Springer, 2016, pp. 21–37.

[21] Z. Shen, Z. Liu, J. Li, Y.-G. Jiang, Y. Chen, X. Xue, Dsod: Learning deeply supervised object detectors from scratch, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 1919–1927.

[22] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, Y. Wei, Deformable convolutional networks, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 764–773.

[23] J. Redmon, A. Farhadi, Yolo9000: better, faster, stronger, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 7263–7271.

[24] A. Farhadi, J. Redmon, Yolov3: An incremental improvement, Computer Vision and Pattern Recognition, cite as (2018).

[25] A. Bochkovskiy, C.-Y. Wang, H.-Y. M. Liao, Yolov4: Optimal speed and accuracy of object detection, arXiv preprint arXiv:2004.10934 (2020).

[26] Z. Cai, N. Vasconcelos, Cascade r-cnn: Delving into high quality object detection, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 6154–6162.

[27] H. Zhang, H. Chang, B. Ma, N. Wang, X. Chen, Dynamic r-cnn: Towards high quality object detection via dynamic training, arXiv preprint arXiv:2004.06002 (2020).

[28] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 2980–2988.

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

[30] H. Law, J. Deng, Cornernet: Detecting objects as paired keypoints, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 734–750.

[31] L. Huang, Y. Yang, Y. Deng, Y. Yu, Densebox: Unifying landmark localization with end to end object detection, arXiv preprint arXiv:1509.04874 (2015).

[32] C. Zhu, Y. He, M. Savvides, Feature selective anchor-free module for single-shot object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 840–849.

[33] Z. Tian, C. Shen, H. Chen, T. He, Fcos: Fully convolutional one-stage object detection, in: Proceedings of the IEEE international conference on computer vision, 2019, pp. 9627–9636.

[34] T. Kong, F. Sun, H. Liu, Y. Jiang, L. Li, J. Shi, Foveabox: Beyound anchor-based object detection, IEEE Transactions on Image Processing 29 (2020) 7389–7398.

[35] J. Yu, Y. Jiang, Z. Wang, Z. Cao, T. Huang, Unitbox: An advanced object detection network, in: Proceedings of the 24th ACM international conference on Multimedia, 2016, pp. 516–520.

[36] X. Zhou, J. Zhuo, P. Krahenbuhl, Bottom-up object detection by grouping extreme and center points, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 850–859.

[37] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, Q. Tian, Centernet: Keypoint triplets for object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 6569–6578.

[38] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, S. Belongie, Feature pyramid networks for object detection, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 2117–2125.

[39] S. Liu, L. Qi, H. Qin, J. Shi, J. Jia, Path aggregation network for instance segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 8759–8768.

[40] S. Liu, D. Huang, Y. Wang, Learning spatial fusion for single-shot object detection, arXiv preprint arXiv:1911.09516 (2019).

[41] G. Ghiasi, T.-Y. Lin, Q. V. Le, Nas-fpn: Learning scalable feature pyramid architecture for object detection, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2019, pp. 7036–7045.

[42] M. Tan, Q. V. Le, Efficientnet: Rethinking model scaling for convolutional neural networks, arXiv preprint arXiv:1905.11946 (2019).

[43] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, arXiv preprint arXiv:1502.03167 (2015).

[44] X. Glorot, A. Bordes, Y. Bengio, Deep sparse rectifier neural networks, in: Proceedings of the fourteenth international conference on artificial intelligence and statistics, 2011, pp. 315–323.

[45] F. Chollet, Xception: Deep learning with depthwise separable convolutions, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1251–1258.

[46] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, A. Zisserman,

The pascal visual object classes (voc) challenge, International journal of computer vision 88 (2) (2010) 303–338.
[47] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, A. Zisserman, The pascal visual object classes challenge: A retrospective, International journal of computer vision 111 (1) (2015) 98–136.
[48] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C. L. Zitnick, Microsoft coco: Common objects in context, in: European conference on computer vision, Springer, 2014, pp. 740–755.
[49] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 3431–3440.
[50] M. Tan, R. Pang, Q. V. Le, Efficientdet: Scalable and efficient object detection, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 10781–10790.
[51] Z. Li, F. Zhou, Fssd: feature fusion single shot multibox detector, arXiv preprint arXiv:1712.00960 (2017).
[52] J. Yi, P. Wu, D. N. Metaxas, Assd: Attentive single shot multibox detector, Computer Vision and Image Understanding 189 (2019) 102827.
[53] C. Zhu, F. Chen, Z. Shen, M. Savvides, Soft anchor-point object detection, arXiv preprint arXiv:1911.12448 (2019).
[54] A. Shrivastava, A. Gupta, R. Girshick, Training region-based object detectors with online hard example mining, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 761–769.
[55] N. Bodla, B. Singh, R. Chellappa, L. S. Davis, Soft-nms–improving object detection with one line of code, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 5561–5569.
[56] G. Ghiasi, Y. Cui, A. Srinivas, R. Qian, T.-Y. Lin, E. D. Cubuk, Q. V. Le, B. Zoph, Simple copy-paste is a strong data augmentation method for instance segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 2918–2928.
[57] B. Singh, M. Najibi, L. S. Davis, Sniper: Efficient multi-scale training, arXiv preprint arXiv:1805.09300 (2018).
[58] Z. Yang, S. Liu, H. Hu, L. Wang, S. Lin, Reppoints: Point set representation for object detection, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 9657–9666.
[59] Y. Li, Y. Chen, N. Wang, Z. Zhang, Scale-aware trident networks for object detection, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 6054–6063.
[60] X. Zhang, F. Wan, C. Liu, X. Ji, Q. Ye, Learning to match anchors for visual object detection, IEEE Transactions on Pattern Analysis and Machine Intelligence (2021).
[61] S. Zhang, C. Chi, Y. Yao, Z. Lei, S. Z. Li, Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 9759–9768.