Corner Proposal Network for Anchor-free, Two-stage Object Detection

Kaiwen Duan[1], Lingxi Xie[2], Honggang Qi[1], Song Bai[3], Qingming Huang[1,4](✉), and Qi Tian[2] (✉)

[1]University of Chinese Academy of Sciences [2]Huawei Inc.
[3]Huazhong University of Science and Technology [4]Peng Cheng Laboratory
duankaiwen17@mails.ucas.ac.cn, 198808xc@gmail.com,
{hgqi,qmhuang}@ucas.ac.cn, songbai.site@gmail.com, tian.qi1@huawei.com

Abstract. The goal of object detection is to determine the class and location of objects in an image. This paper proposes a novel anchor-free, two-stage framework which first extracts a number of object proposals by finding potential corner keypoint combinations and then assigns a class label to each proposal by a standalone classification stage. We demonstrate that these two stages are effective solutions for improving recall and precision, respectively, and they can be integrated into an end-to-end network. Our approach, dubbed Corner Proposal Network (CPN), enjoys the ability to detect objects of various scales and also avoids being confused by a large number of false-positive proposals. On the MS-COCO dataset, CPN achieves an AP of 49.2% which is competitive among state-of-the-art object detection methods. CPN also fits the scenario of computational efficiency, which achieves an AP of 41.6%/39.7% at 26.2/43.3 FPS, surpassing most competitors with the same inference speed. Code is available at https://github.com/Duankaiwen/CPNDet.

Keywords: Object Detection, Anchor-Free Detector, Two-stage Detector, Corner Keypoints, Object Proposals

1 Introduction

Powered by the rapid development of deep learning [21], in particular deep convolutional neural networks [18,35,13], researchers have designed effective algorithms for object detection [11]. This is a challenging task since objects can appear in any scale, shape, and position in a natural image, yet the appearance of objects of the same class can be very different.

The two keys to a detection approach are to find objects with different kinds of geometry (*i.e.*, high recall) as well as to assign an accurate label to each detected object (*i.e.*, high precision). Existing object detection approaches are roughly categorized according to how objects are located and how their classes are determined. For the first issue, early research efforts are mostly **anchor-based** [10,34,3,6,27,24,46], which involved placing a number of size-fixed bounding boxes on the image plane, while this methodology was later challenged by

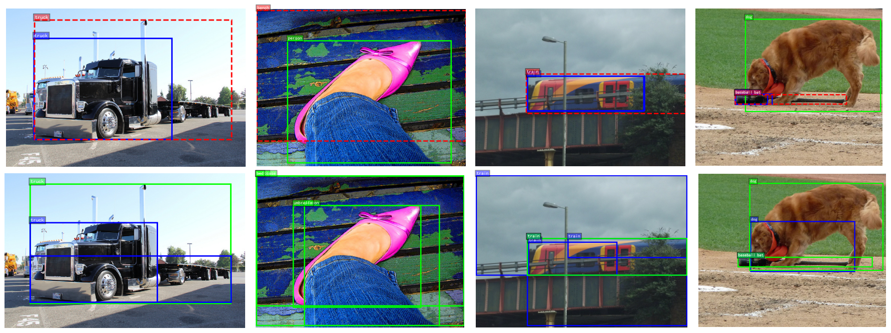

Fig. 1. Typical errors by existing object detection approaches (best viewed in color). **Top**: an anchor-based method (*e.g.*, Faster R-CNN [34]) may have difficulty in finding objects with a peculiar shape (*e.g.*, with a very large size or an extreme aspect ratio). **Bottom**: an anchor-free method (*e.g.*, CornerNet [19]) may mistakenly group irrelevant keypoints into an object. Green, blue and red bounding-boxes indicate true positives, false positives and false negatives, respectively.

anchor-free [19,8,17,42,50,39] methods which suggested depicting each object with one or few keypoints and the geometry. These potential objects are named proposals, and for each of them, the class label is either inherited from a previous output or verified by an individual classifier trained for this purpose. This brings a debate between the so-called **two-stage** and **one-stage** approaches, in which people tend to believe that the former works slower but produces higher detection accuracy.

This paper provides an alternative opinion on the design of object detection approaches. There are two arguments. **First**, the recall of a detection approach is determined by its ability to locate objects of different geometries, especially those with rare shapes, and anchor-free methods (in particular, the methods based on locating the border of objects) are potentially better in this task. **Second**, anchor-free methods often incur a large number of false positives, and thus an individual classifier is strongly required to improve the precision of detection, see Fig. 1. Therefore, we inherit the merits of both anchor-free and two-stage object detectors and design an efficient, end-to-end implementation.

Our approach is named **Corner Proposal Network** (CPN). It detects an object by locating the top-left and bottom-right corners of it and then assigning a class label to it. We make use of the keypoint detection method of CornerNet [19] but, instead of grouping the keypoints with keypoint feature embedding, enumerate all valid corner combinations as potential objects. This leads to a large number of proposals, most of which are false positives. We then train a classifier to discriminate real objects from incorrectly paired keypoints based on the corresponding regional features. There are two steps for classification, with the first one, a binary classifier, filtering out a large part of proposals (*i.e.*, that do not correspond to objects), and the second one, with stronger features, re-ranking the survived proposals with multi-class classification scores.

The effectiveness of CPN is verified on the MS-COCO dataset [25], one of the most challenging object detection benchmarks. Using a 104-layer stacked Hourglass network [30] as the backbone, CPN reports an AP of 49.2%, which outperforms the previously best anchor-free detectors, CenterNet [8], by a significant margin of 2.2%. In particular, CPN enjoys even larger accuracy gain in detecting objects with peculiar shapes (*e.g.*, very large or small areas or extreme aspect ratios), demonstrating the advantage of using anchor-free methods for proposal extraction. Last but not least, CPN can also fit scenarios that desire for network efficiency. Working on a lighter backbone, DLA-34 [43], and switching off image flip in inference, CPN achieves an AP of 41.6% at 26.2 FPS or 39.7% at 43.3 FPS, surpassing most competitors with the same inference speed.

2 Related Work

Object detection is an important yet challenging task in computer vision. It aims to obtain a tight bounding-box as well as a class label for each object in an image. In recent years, with the rapid development of deep learning, most powerful object detection methods are based on training deep neural networks [11,10]. According to the way of localizing objects, existing detection approaches can be roughly categorized into anchor-based and anchor-free methods.

An **anchor-based approach** starts with placing a large number of anchors, which are regional proposals with different but fixed scales and shapes, and are uniformly distributed on the image plane. These anchors are then considered as object proposals and an individual classifier is trained to determine the objectness as well as the class of each proposal [34]. Beyond this framework, researchers made efforts in two aspects, namely, improving the basic quality of regional features extracted from the proposal, and arriving at a better alignment between the proposals and features. For the first type of efforts, typical examples include using more powerful network backbones [13,36,14] and using hierarchical features to represent a region [23,34,27]. Regarding the second type, there exist methods to align anchors to features [46,47], align features to anchors [7,5], and adjust the anchors after classification has been done [34,27,3].

Alternatively, an **anchor-free approach** does not assume the objects to come from uniformly distributed anchors. Early efforts including DenseBox [15] and UnitBox [44] proved that the detectors can achieve the detection task without anchors. Recently, anchor-free approaches have been greatly promoted by the development of keypoint detection [29] and the assist of the focal loss [24]. The fundamental of anchor-free approaches is usually one or few keypoints. Depending on how keypoints are used for object depiction, anchor-free approaches can be roughly categorized into point-grouping detectors and point-vector detectors. Point-grouping detectors, including CornerNet [19], CenterNet [8], ExtremeNet [49], *etc,* group more than one keypoints into an object, while the point-vector detectors such as FCOS [39], CenterNet [48], FoveaBox [17], SAPD [50], *etc.*, use a keypoint and a vector of object geometry (*e.g.*, the width, height, or its distance to the borders) to determine the shape of objects.

Based on the object proposals, it remains to determine whether each proposal is an object and what class of object it is. There is also discussion on using **two-stage** and **one-stage** detectors for object detection. A two-stage detector [34,12,3,31,22] refers to an individual classifier is trained for this purpose, while a one-stage detector [33,27,24,46,5] mostly uses classification cues from the previous stage. Two-stage detectors are often more accurate but slower, compared to one-stage detectors. To accelerate it, an efficient method is to partition classification into two steps [6,23,12,3,31], with the first step filtering out most easy judged false positives, and the second step using heavier computation to assign each survived proposal a class label.

3 Our Approach

Object detection starts with an image $\mathbf{I}$ denoted by raw pixels, on which a few rectangles, often referred to as bounding-boxes, that tightly covers the objects are labeled with a class label. Denote a ground-truth bounding-box as $\mathbf{b}_n^\star$, $n = 1, 2, \ldots, N$, the corresponding class label as $c_n^\star$, and $\mathbf{I}_n^\star$ represents the image region within the corresponding bounding-box. The goal is to locate a few bounding-boxes, $\mathbf{b}_m$, $m = 1, 2, \ldots, M$, and assign each one with a class label, c_m, so that the sets of $\{\mathbf{b}_n^\star, c_n^\star\}_{n=1}^N$ and $\{\mathbf{b}_m, c_m\}_{m=1}^M$ are as close as possible.

3.1 Anchor-based or Anchor-free? One-stage or Two-stage?

We focus on two important choices of object detection, namely, whether to use anchor-based or anchor-free methods for proposal extraction, and whether to use one-stage or two-stage methods for determining the class of proposals. Based on these discussions, we present a novel framework in the next subsection.

We first investigate anchor-based *vs.* anchor-free methods. Anchor-based methods first place a number of anchors on the image as object proposals and then use an individual classifier to judge the objectness and class of each proposal. Most often, each anchor is associated with a specific position on the image and its size is fixed, although the following process named bounding-box regression can slightly change its geometry. Anchor-free methods do not assume the objects to come from anchors of relatively fixed geometry, and instead, locate one or few keypoints of an object and determine its geometry and/or class afterward.

Our core opinion is that **anchor-free methods have better flexibility of locating objects with arbitrary geometry, and thus a higher recall.** This is mainly due to the design nature of anchors, which is mostly empirical (*e.g.*, to reduce the number of anchors and improve efficiency, only common object sizes and shapes are considered), the detection algorithm is potential of lower flexibility and objects with a peculiar shape can be missing. Typical examples are shown in Fig. 1, and a quantitative study is provided in Table 1. We evaluate four object proposal extraction methods as well as our work on the MS-COCO validation dataset and show that anchor-free methods often have a higher overall recall, which is mainly due to their advantages in two scenarios. **First**, when the

Table 1. Comparison among the average recall (AR) of anchor-based and anchor-free detection methods. Here, the average recall is recorded for targets of different aspect ratios and different sizes. To explore the limit of the average recall for each method, we exclude the impacts of bounding-box categories and sorts on recall, and compute it by allowing at most 1000 object proposals. AR_{1+}, AR_{2+}, AR_{3+} and AR_{4+} denote box area in the ranges of $(96^2, 200^2]$, $(200^2, 300^2]$, $(300^2, 400^2]$, and $(400^2, +\infty)$, respectively. 'X' and 'HG' stand for ResNeXt and Hourglass, respectively.

Method	Backbone	AR	AR_{1+}	AR_{2+}	AR_{3+}	AR_{4+}	$AR_{5:1}$	$AR_{6:1}$	$AR_{7:1}$	$AR_{8:1}$
Faster R-CNN [34]	X-101-64x4d	57.6	73.8	77.5	79.2	86.2	43.8	43.0	34.3	23.2
FCOS [39]	X-101-64x4d	64.9	82.3	87.9	89.8	95.0	45.5	40.8	34.1	23.4
CornerNet [19]	HG-104	66.8	85.8	92.6	95.5	98.5	50.1	48.3	40.4	36.5
CenterNet [8]	HG-104	66.8	87.1	93.2	95.2	96.9	50.7	45.6	40.1	32.3
CPN (this work)	HG-104	68.8	88.2	93.7	95.8	99.1	54.4	50.6	46.2	35.4

object is very large, *e.g.*, larger than 400^2 pixels, Faster R-CNN, an anchor-based approach, does not report a much higher recall. This is not expected because large objects should be easier to detect, as the other three anchor-free methods suggest. **Second**, Faster R-CNN suffers a very low recall when the aspect ratio of the object becomes peculiar, *e.g.*, $5:1$ and $8:1$, in which cases the recalls are significantly lower than CornerNet [19] and CenterNet [8], because no pre-defined anchors (also used in other variants [3,27,31]) can fit these objects. A similar phenomenon is also observed in FCOS, an anchor-free approach which represents an object by a keypoint and the distance to the border, because it is difficult to predict an accurate distance when the border is far from the center. Since CornerNet and CenterNet group corner (and center) keypoints into an object, they somewhat get rid of this trouble. Therefore, we choose anchor-free methods, in particular, **point-grouping** methods (CornerNet and CenterNet), to improve the recall of object detection. Moreover, we report the corresponding results of CPN, the method proposed in this paper, which demonstrates that CPN inherits the merits of CenterNet and CornerNet and has better flexibility of locating objects, especially with peculiar shapes.

Table 2. Anchor-free detection methods such as CornerNet and CenterNet suffer a large number of false positives and can benefit from incorporating richer semantics for judgment. Here, AP_{original}, AP_{refined}, and AP_{correct} indicate the AP of the original output, after non-object proposals are removed, and after the correct label is assigned to each survived proposal. Both AP_{refined} and AP_{correct} require ground-truth labels.

Method	Backbone	AP_{original}	AP_{refined}	AP_{correct}	$AR^{100}_{\text{original}}$	$AR^{100}_{\text{refined}}$	$AR^{100}_{\text{correct}}$
CornerNet [19]	HG-52	37.6	53.8	60.3	56.7	65.3	69.2
CornerNet [19]	HG-104	41.0	56.6	61.6	59.1	67.0	70.4
CenterNet [8]	HG-52	41.3	51.9	56.6	59.5	61.1	64.2
CenterNet [8]	HG-104	44.8	55.3	59.9	62.2	65.1	68.4

However, anchor-free methods free the constraints of finding object proposals, it encounters a major difficulty of building a close relationship between keypoints and objects, since the latter often requires richer semantic information. As shown in Fig. 1, lacking semantics can incur a large number of false positives and thus harms the precision of detection. We take CornerNet [19] and CenterNet [8] with potentially high recalls as examples. As shown in Table 2, the CornerNets with 52-layer and 104-layer Hourglass networks achieved APs of 37.6% and 41.0% on the MS-COCO validation dataset, while many of the detected 'objects' are false positives. Either when we remove the non-object proposals or assign each preserved proposal with a correct label, the detection accuracy goes up significantly. This observation also holds on CenterNet [8], which added a center point to filter out false positives but obviously did not remove them all. To further alleviate this problem, we need to inherit the merits of two-stage methods, which extract the features within proposals and train a classifier to filter out false positives.

3.2 The Framework of Corner Proposal Network

Motivated by the above, the goal of our approach is to integrate the advantages of anchor-free methods and alleviate their drawbacks by leveraging the mechanism of discrimination from two-stage methods. We present a new framework named **Corner-Proposal-Network** (CPN). It uses an anchor-free method to extract object proposals followed by efficient regional feature computation and classification to filter out false positives. Fig. 2 shows the overall framework which contains two stages, and details of the two stages are elaborated as follows.

- **Stage 1: Anchor-free Proposals with Corner Keypoints**

The first stage is an anchor-free proposal extraction process, in which we assume that each object is located by two keypoints determining its top-left and bottom-right corners. We follow CornerNet [19] to locate an object with a pair of keypoints located in its top-left and bottom-right corners, respectively. For each class, we compute two heatmaps (*i.e.*, the top-left heatmap and the bottom-right heatmap, each value on a heatmap indicates the probability that a corner keypoint occurs in the corresponding position) with a 4×-reduced resolution compared to the original image. The heatmaps are equipped with two loss terms, namely, a focal loss $\mathcal{L}_{\mathrm{det}}^{\mathrm{corner}}$ to locate the keypoint on the heatmap and a offset loss $\mathcal{L}_{\mathrm{offset}}^{\mathrm{corner}}$ to learn its offset to the accurate corner position. Then, a fixed number of keypoints (K top-left and K bottom-right) are extracted from all heatmaps. This implies that each corner keypoint is equipped with a class label.

Next, each valid pair of keypoints defines an object proposal. Here by valid we mean that two keypoints belong to the same class (*i.e.*, extracted from the top-left heatmap and the bottom-right heatmap of the same class), and the x and y coordinates of the top-left point are smaller than that of the bottom-right point, respectively. This leads to a large number of false positives (incorrectly paired corner keypoints) on each image, and we leave the task of discriminating and classifying these proposals in the second stage.

As a side comment, we emphasize that although we extract object proposals based on CornerNet, the follow-up mechanism of determining objectness and

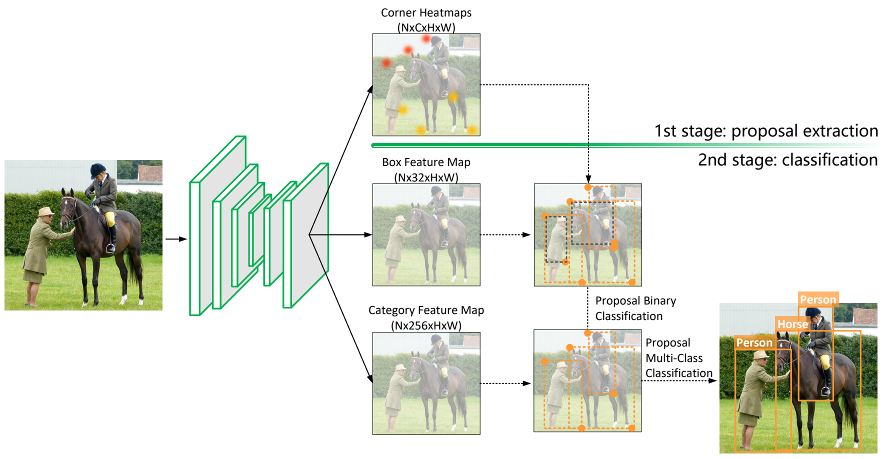
Corner Heatmaps (NxCxHxW)
1st stage: proposal extraction
2nd stage: classification
Box Feature Map (Nx32xHxW)
Category Feature Map (Nx256xHxW)
Proposal Binary Classification
Proposal Multi-Class Classification
Person
Horse
Person

Fig. 2. The overall framework of proposed CPN. It first predicts corners to compose a number of object proposals, and then applies two-step classification to filter out false positives and assign a class label for each survived proposal.

class is quite different. CornerNet generates objects by projecting the keypoints to a one-dimensional space, and grouping keypoints with closely embedded numbers into the same instance. We argue that the embedding process, while necessary under the assumption that no additional computation can be used, can incur significant errors in pairing keypoints. In particular, there is no guarantee that the embedding function (assigning a number to each object) is learnable, and more importantly, the loss function only works in each training image to force the embedded numbers of different objects to be separated, but this mechanism often fails to generalize to unseen scenarios, *e.g.*, even when multiple training images are simply concatenated together, the embedding function that works well on separate images can fail dramatically. Differently, our method determines object instances using an individual classifier, which makes full use of the internal features to improve accuracy. Please refer to Table 6 for the advantage of an individual classifier over instance embedding.

- **Stage 2: Two-step Classification for Filtering Proposals**

Thanks to the high resolution of the keypoint heatmap and a flexible mechanism of grouping keypoints, the detected objects can be of an arbitrary size, and the upper-bound of recall is largely improved. However, this strategy increases the number of proposals. For example, we follow CenterNet [8] to set K to be 70, which results in an average of 2,500 object proposals on each image. Most of these proposals are false positives, and individually validating and classifying each of them is computationally expensive. To solve this issue, an efficient, **two-step** classification method is designed for the second stage, which first removes 80% of proposals with a light-weighted binary classifier, and then applies a fine-level classifier to determine the class label of each survived proposal.

The **first step** involves training a binary classifier to determine whether each proposal is an object. We first adopt RoIAlign [12] with a kernel size of 7×7 to extract the features for each proposal on the box feature map (see Fig. 2). Then, a $32 \times 7 \times 7$ convolution layer is followed to obtain the classification score for each proposal. A binary classifier is built, with the loss function being:

$$\mathcal{L}_{\text{prop}} = -\frac{1}{N} \sum_{m=1}^{M} \begin{cases} (1 - p_m)^{\alpha} \log (p_m), & \text{if } \mathrm{IoU}_m \geqslant \tau \\ p_m^{\alpha} \log (1 - p_m), & \text{otherwise} \end{cases}, \tag{1}$$

where M is the total number of object proposals, N denotes the number of positive samples, p_m denotes the objectness score for the m-th proposal, $p_m \in [0, 1]$, and IoU_m denotes the maximum IoU value between the m-th proposal and all the ground-truth bounding-boxes. τ is the IoU threshold, set to be 0.7 throughout this paper. This is to sample a few positive examples to avoid training data imbalance [24]. $\alpha = 2$ is a hyper-parameter that smoothes the loss function. According to [24], we use $\pi = 0.1$, so the value of the biases is around -2.19.

The **second step** follows to assign a class label for each survived proposal. This step is very important, since the class labels associated to the corner keypoints are not always reliable. Although we rely on the corner classes to reject invalid corner pairs, the consensus between them may be incorrect due to the lack of information from the ROI regions, so we need a more powerful classifier that incorporates the ROI features to make the final decision. To this end, we train another classifier with C outputs where C is the number of classes in the dataset. This classifier is also built upon the RoIAlign-features extracted in the first step, but instead extract the features from the category feature map (see Fig. 2) to preserve more information and a C dimensional vector is obtained using a $256 \times 7 \times 7$ convolution layer, for each of the survived proposals. Then, a C-way classifier is built with a similar loss function considering the class label:

$$\mathcal{L}_{\text{class}} = -\frac{1}{\hat{N}} \sum_{m=1}^{\hat{M}} \sum_{c=1}^{C} \begin{cases} (1 - q_{m,c})^{\beta} \log (q_{m,c}), & \text{if } \mathrm{IoU}_{m,c} \geqslant \tau \\ q_{m,c}^{\beta} \log (1 - q_{m,c}), & \text{otherwise} \end{cases}, \tag{2}$$

where $\hat{M}$ and $\hat{N}$ denote the number of survived proposals and the number of positive samples within them, respectively. $\mathrm{IoU}_{m,c}$ denotes the maximum IoU value between the m-th proposal and all the ground-truth bounding-boxes of the c-th class, and the IoU threshold, τ, remains unchanged. $q_{m,c}$ is the classification score for the c-th class of the m-th object, and β plays a similar role as α, and we also fix it to be 2 in this paper.

Here we emphasize the differences between DeNet [40] and our method, although they are similar in the idea level. First, we equip each corner keypoint with a multi-class label rather than a binary label, thus we can make use of the class labels to reject the unnecessary invalid corner pairs to save the computational costs of the overall framework. Second, we use an extra lightweight binary classification network to first reduce the number of proposals to be processed by the classification network, which improves the efficiency of our approach, while

DeNet used one-step classification. Finally, we design a novel variant of the focal loss for the two classifiers, which is different from the maximum likelihood function in DeNet. This is mainly designed to solve the significant imbalance between the positive and negative proposals during the training process.

3.3 The Inference Process

The inference process simply repeats the training process but uses thresholds to filter out clearly low-quality proposals. Note that even with augmented positive training data, the predicted scores, p_m and $q_{m,c}$, are biased towards 0. So, in the inference stage, we use a relatively low threshold (0.2 in this paper) in the first step to allow more proposals to survive. For each proposal, provided the RoIAlign-features, the computational cost of the first classifier is about 10% of that of the second one. Under the threshold of 0.2, the average fraction of survived proposals is around 20%, making the overheads of these two stages comparable.

For each proposal survived to the second step, we assign it with up to 2 class labels, corresponding to the dominant class of the corner keypoints and that of the proposal (two classes may be identical, if not, the proposal becomes two proposals with potentially different scores). For each candidate class, we denote s_1 as the corner classification score (the average of two corner keypoints, in the range of $(0, 1)$), and s_2 as the regional classification score (the probability of the proposal class label, predicted by the multi-class classifier, also in the range of $(0, 1)$). We assume that both scores contribute to the final score, and a positive evidence should be added if either score is larger than 0.5. Therefore, we compute the score by $s_c = (s_1 + 0.5)\,(s_2 + 0.5)$, then we will apply normalization to rescale this score into the $[0, 1]$. We finally preserve 100 proposals with highest scores into evaluation. In Table 4, we will show that two classifiers provide complementary information and boost the detection accuracy.

4 Experiments

4.1 Dataset and Evaluation Metrics

We evaluate our approach on the MS-COCO dataset [25], one of the most popular object detection benchmarks. It contains a total of 120K images with more than 1.5 million bounding boxes covering 80 object categories, making it a very challenging dataset. Following the common practice [24,23], we train our model using the 'trainval35k', which is the union set of 80K training images and 35K (a subset of) validation images. We report evaluation results on the standard test-dev set, which has no public annotations, by uploading the results to the evaluation server. For the ablation studies and visualization experiments, we use the 5K validation images remained in the validation set.

We use the average precision (AP) metric defined in MS-COCO to characterize the performance of our approach as well as other competitors. AP computes the average precision over ten IoU thresholds (*i.e.*, 0.5 : 0.05 : 0.95), for all categories. Meanwhile, we follow the convention to report some other important metrics, *e.g.*, AP_{50} and AP_{75} are computed at single IoU thresholds of 0.50 and 0.75 [9], and $\mathrm{AP}_{\mathrm{small}}$, $\mathrm{AP}_{\mathrm{medium}}$, and $\mathrm{AP}_{\mathrm{large}}$ are computed under different object scales (*i.e.*, small for area $< 32^2$, medium for $32^2 <$ area $< 96^2$, and large for area $> 96^2$), respectively. All metrics are computed by allowing at most 100 proposals preserved on each testing image.

4.2 Implementation Details

We implement our method using Pytorch [32], and refer to some codes from CornerNet [19], mmdetection [4] and CenterNet [48]. We use CornerNet [19] and CenterNet [8] as our baselines. The stacked Hourglass networks [30] with 52 and 104 layers are trained for keypoint extraction, with all modifications made by CornerNet preserved. In addition, we experiment another backbone named DLA-34 [43]. We follow the modifications made by CenterNet [48], but replace the deformable convolutional layers with normal layers.

Training. All networks are trained from scratch, except the DLA-34, which is initialized with ImageNet pretrain. Cascade corner pooling [8] is used to help the network better detect corners. The input image is resized into 511×511, and the output resolutions for the four feature maps (the top-left and bottom-right heatmaps, the proposal and class feature maps) are all 128×128. The data augmentation strategy presented in [19] is used. The overall loss function is

$$\mathcal{L} = \mathcal{L}_{\mathrm{det}}^{\mathrm{corner}} + \mathcal{L}_{\mathrm{offset}}^{\mathrm{corner}} + \mathcal{L}_{\mathrm{prop}} + \mathcal{L}_{\mathrm{class}}, \tag{3}$$

which we use an Adam [16] optimizer to train our model. On eight NVIDIA Tesla-V100 (32GB) GPUs, we use a batch size of 48 (6 samples on each card) and train the model for 200K iterations with a base learning rate of 2.5×10^{-4} followed by another 50K iterations with a reduced learning rate of 2.5×10^{-5}. The training lasts about 9 days, 5 days and 3 days for the backbones of Hourglass-104, Hourglass-52 and DLA-34, respectively.

Inference. Following [19], both single-scale and multi-scale detection processes are performed. For single-scale testing, we feed the image with the original resolution into the network, while for multi-scale testing, the image is resized into different resolutions (0.6×, 1×, 1.2×, 1.5×, and 1.8×) and then fed into the network. Flip argumentation is added to both single-scale or multi-scale evaluation. For multi-scale evaluation, the predictions for all scales (including the flipped variants) are fused into the final result. we use soft-NMS [2] to suppress the redundant bounding-boxes, and preserve 100 top-scored boxes for evaluation.

4.3 Comparisons with State-of-the-Art Detectors

We report the inference accuracy of CPN on the MS-COCO test-dev set and compare with the state-of-the-art detectors, as shown in Table 3. CPN obtains a significant improvement compared to CornerNet [19] and CenterNet [8], two

Table 3. Inference accuracy (%) of CPN and state-of-the-art detectors on the COCO *test-dev* set. CPN ranks among the top of state-of-the-art detectors. 'R', 'X', 'HG', 'DCN' and '†' denote ResNet, ResNeXt, Hourglass, Deformable Convolution Network [7], and multi-scale training or testing, respectively.

Method	Backbone	Input Size	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Anchor-based:								
Faster R-CNN [23]	R-101	600	36.2	59.1	39.0	18.2	39.0	48.2
RetinaNet [24]	R-101	800	39.1	59.1	42.3	21.8	42.7	50.2
Cascade R-CNN [3]	R-101	800	42.8	62.1	46.3	23.7	45.5	55.2
Libra R-CNN [31]	X-101-64x4d	800	43.0	64.0	47.0	25.3	45.6	54.6
Grid R-CNN [28]	X-101-64x4d	800	43.2	63.0	46.6	25.1	46.5	55.2
YOLOv4 [1]	CSPDarknet-53	608	43.5	65.7	47.3	26.7	46.7	53.3
AlignDet [5]	X-101-32x8d	800	44.1	64.7	48.9	26.9	47.0	54.7
AB+FSAF [51] †	X-101-64x4d	800	44.6	65.2	48.6	29.7	47.1	54.6
FreeAnchor [47] †	X-101-32x8d	≤1280	47.3	66.3	51.5	30.6	50.4	59.0
PANet [26] †	X-101-64x4d	840	47.4	67.2	51.8	30.1	51.7	60.0
TridentNet [22] †	R-101-DCN	800	48.4	69.7	53.5	31.8	51.3	60.3
ATSS [45] †	X-101-64x4d-DCN	800	50.7	68.9	56.3	33.2	52.9	62.4
EfficientDet [38]	EfficientNet [37]	1536	**53.7**	**72.4**	**58.4**	-	-	-
Anchor-free:								
GA-Faster-RCNN [41]	R-50	800	39.8	59.2	43.5	21.8	42.6	50.7
FoveaBox [17]	R-101	800	42.1	61.9	45.2	24.9	46.8	55.6
ExtremeNet [49] †	HG-104	≤1.5×	43.2	59.8	46.4	24.1	46.0	57.1
FCOS [39] w/ imprv	X-101-64x4d	800	44.7	64.1	48.4	27.6	47.5	55.6
CenterNet [48] †	HG-104	≤1.5×	45.1	63.9	49.3	26.6	47.1	57.7
RPDet [42] †	R-101-DCN	800	46.5	67.4	50.9	30.3	49.7	57.1
SAPD [50]	X-101-64x4d	800	45.4	65.6	48.9	27.3	48.7	56.8
SAPD [50]	X-101-64x4d-DCN	800	47.4	67.4	51.1	28.1	50.3	61.5
CornerNet [19]	HG-104	ori.	40.5	56.5	43.1	19.4	42.7	53.9
CornerNet [19] †	HG-104	≤1.5×	42.1	57.8	45.3	20.8	44.8	56.7
CenterNet [8]	HG-104	ori.	44.9	62.4	48.1	25.6	47.4	57.4
CenterNet [8] †	HG-104	≤1.8×	47.0	64.5	50.7	28.9	49.9	58.9
CPN	DLA-34	ori.	41.7	58.9	44.9	20.2	44.1	56.4
CPN	HG-52	ori.	43.9	61.6	47.5	23.9	46.3	57.1
CPN	HG-104	ori.	47.0	65.0	51.0	26.5	50.2	60.7
CPN †	DLA-34	≤1.8×	44.5	62.3	48.3	25.2	46.7	58.2
CPN †	HG-52	≤1.8×	45.8	63.9	49.7	26.8	48.4	59.4
CPN †	HG-104	≤1.8×	**49.2**	**67.4**	**53.7**	**31.0**	**51.9**	**62.4**

direct baselines. Specifically, CPN-52 (indicating that the backbone is Hourglass-52) reports a single-scale testing AP of 43.9% and a multi-scale testing AP of 45.8%, which surpasses CornerNet-104, with a deeper backbone, by 3.4% and 3.7%, respectively. The best performance of CPN reaches an AP of 49.2%, surpassing all published anchor-free approaches to the best of our knowledge. Meanwhile, CPN is also competitive among anchor-based detectors, *e.g.*, CPN-52 reports a single-scale testing AP of 43.9% which is comparable to 44.1% of

Table 4. The detection performance (%) of different classification options on CPN.

Backbone	B-Classifier	M-Classifier	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	AR_{100}
HG-52			38.6	54.5	41.4	19.0	40.6	54.4	57.4
	✓		42.0	59.7	45.1	24.2	45.0	56.6	60.5
		✓	42.4	60.2	45.7	23.3	44.7	58.6	59.0
	✓	✓	**43.8**	**61.4**	**47.4**	**24.7**	**46.5**	**60.0**	**60.9**

AlignDet [5], and CPN-104 reports a single-scale testing AP of 47.0% which is comparable 47.4% of PANet [26].

CPN also takes the lead in other metrics. For example, AP_{50} and AP_{75} reflect the accuracy of proposal localization and class prediction. Compared to CenterNet, CPN reports higher AP scores especially for AP_{75} (*e.g.*, CPN-104 reports a single-scale testing AP_{75} of 51.0%, claiming an improvement of 2.9% over CenterNet). This suggests that some inaccurate bounding boxes with IoU value between 0.5 and 0.7 are difficult for CenterNet to filter out with merely center information incorporated. AP_S, AP_M and AP_L reflect the detection accuracy for objects with different scales. CPN improves more for AP_M and AP_L than AP_S (*e.g.*, CPN-104 reports single-scale testing AP_S, AP_M and AP_L of 26.5%, 50.2% and 60.7%, which improves by 0.9%, 2.8% and 3.3% from CenterNet, respectively). This is because medium and large objects require richer semantic information to be extracted from the proposal, which is not likely to be handled well with a center keypoint.

4.4 Classification Improves Precision

We investigate the improvement of precision brought by the classification stage. Note that there are two classifiers, with the first one (binary) determines the objectness of each proposal, and the second one (multi-class) providing complementary information of the class label. We perform ablation study in Table 4 to analyze the contribution of individual classifiers – in the scenarios that the multi-class classifier is missing, we directly use the class label of the corner keypoints as the final prediction. On the one hand, both classifiers can improve the AP of the basic model (one-stage corner keypoint grouping) significantly (3.4% and 3.8% absolute gains). On the other hand, two classifiers provide complementary information, so that combining them can further improve the AP by more than 1%. This indicates that the semantic information required for determining the objectness and class label of a proposal is slightly different.

We have discussed the importance of the incorrect bounding box suppression for the improvement of detection accuracy and recall in Section 3.1. For a more intuitive analysis of false positives, we adopt the average false discovery rate (AF) metric[1] [8] to quantify the fraction of incorrectly grouped proposals for

[1] $AF = 1 - \widetilde{AP}$, where $\widetilde{AP}$ is computed over IoU thresholds of $[0.05 : 0.05 : 0.5]$ for all categories. $AF_\tau = 1 - \widetilde{AP}_\tau$, where $\widetilde{AP}_\tau$ is computed at the IoU threshold of τ%, $AF_{\text{scale}} = 1 - \widetilde{AP}_{\text{scale}}$, where scale $\in$ {small, medium, large} indicates the object size.

Table 5. We report the average false discovery rates (%, lower is better) for CornerNet, CenterNet and CPN on the MS-COCO validation dataset. The results show that our approach generates fewer false positives. Under the same corner keypoint extractor, this is the key to outperform the baselines in the AP metrics.

Method	Backbone	AF	AF_5	AF_{25}	AF_{50}	AF_S	AF_M	AF_L
CornerNet [19]	HG-52	40.4	35.2	39.4	46.7	62.5	36.9	28.0
CenterNet [8]	HG-52	35.1	30.7	34.2	40.8	53.0	31.3	24.4
CPN	HG-52	**33.4**	**29.5**	**32.5**	**38.6**	**52.0**	**29.2**	**21.0**
CornerNet [19]	HG-104	37.8	32.7	36.8	43.8	60.3	33.2	25.1
CenterNet [8]	HG-104	32.4	28.2	31.6	37.5	50.7	27.1	23.0
CPN	HG-104	**30.6**	**26.9**	**29.7**	**35.5**	**48.8**	**25.7**	**19.2**

Table 6. The detection performance (%) of using different ways (instance embedding and binary classification) to determine the validity of a proposal.

Method	Backbone	Objectness	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	AR_{100}
CornerNet [19]	HG-52	Embedding	38.3	54.2	40.5	18.5	39.6	52.2	56.7
		B-Classifier	**42.3**	**59.5**	**45.4**	**24.6**	**45.4**	**57.6**	**59.9**
CenterNet [8]	HG-52	Embedding	41.3	59.2	43.9	23.6	43.6	53.6	59.0
		B-Classifier	**42.6**	**59.8**	**45.8**	**25.1**	**45.7**	**57.7**	**60.1**

different detectors. Results are shown in Table 5. CPN-52 and CPN-104 report AF of 33.4% and 30.6%, respectively, which are lower than the direct baselines, CornerNet and CenterNet.

Note that these three methods share a similar way of extracting corner keypoints, but CornerNet suffers large AF values due to the lack of validation beyond the proposals. CenterNet, by forcing a center keypoint to be detected, was believed effective in filtering out false positives, and our approach, by reevaluating the proposal based on regional features, performs better than CenterNet. More importantly, by inserting an individual classification stage, CPN alleviates the false positives produced by instance embedding, as shown in Table 6.

4.5 Inference Speed

To show that CPN can generate high-quality bounding boxes with small computational costs, we report the inference speed for CPN on the MS-COCO validation dataset under different settings and compare the results to state-of-the-art efficient detectors, as shown in Table 7. For fair comparison, we test the inference speed for all detectors on an NVIDIA Tesla-V100 GPU. CPN-104 reports an FPS/AP of 7.3/46.8%, which is both faster and better than CenterNet-104 (5.1/44.8%) under the same setting. With a lighter backbone of Hourglass-52, CPN-52 reports an FPS/AP of 9.9/43.8%, which outperforms both CornerNet-52 and CenterNet-52. This indicates that two-stage detectors are not necessarily slow – our solution, by sharing a large amount of computation between the first

Table 7. Inference speed of CPN under different conditions *vs.* other detectors on the MS-COCO validation dataset. FPS is measured on the on an NVIDIA Tesla-V100 GPU. CPN achieves a good trade-off between accuracy and speed.

Method	Backbone	Input Size	Flip	AP	FPS
FCOS [39]	X-101-64x4d	800	×	42.6	8.1
Faster R-CNN [34]	X-101-64x4d	800	×	41.1	8.2
CornerNet-lite [20]	HG-Squeeze	ori.	✓	36.5	22.0
CornerNet [19]	HG-104	ori.	✓	41.0	5.8
CenterNet [8]	HG-104	ori.	✓	44.8	5.1
CPN	HG-104	ori.	✓	46.8	7.3
CPN	HG-52	ori.	✓	43.8	9.9
CPN	HG-104	0.7×ori.	×	40.5	17.9
CPN	DLA-34	ori.	✓	41.6	26.2
CPN	DLA-34	ori.	×	39.7	43.3

(for keypoint extraction) and the second (for feature extraction) stage, achieves a good trade-off between inference speed and accuracy.

In the scenarios that require faster inference speed, CPN can be further accelerated by replacing with a lighter backbone and not using flip augmentation at the inference stage. For example, with the backbone of DLA-34 [43], CPN reports FPS/AP of 43.3/39.7% and 26.2/41.6% with and without flip augmentation, surpassing other competitors with the similar computational cost.

5 Conclusions

In this paper, we present an anchor-free, two-stage object detection framework. It starts with extracting corner keypoints and composing them into object proposals, and applies two-step classification to filter out false positives. With the above two stages, the recall and precision of detection are significantly improved, and the final result ranks among the top of existing object detection methods. We have also achieved a satisfying trade-off between accuracy and complexity.

The most important take-away is that anchor-free methods are more flexible in proposal extraction, while an individual discrimination stage is required to improve precision. When implemented properly, such a two-stage framework can be efficient in evaluation. Therefore, the debate on using one-stage or two-stage detectors seems not critical, but the importance of accurately localizing and recognizing instances becomes more significant.

Acknowledgments

This research was supported in part by National Natural Science Foundation of China: 61620106009, 61931008, U1636214 and 61836002, and Key Research Program of Frontier Sciences, CAS: QYZDJ-SSWSYS013. We thank Dr. Wei Zhang and Jianzhong He for instructive discussions.

References

1. Bochkovskiy, A., Wang, C.Y., Liao, H.Y.M.: Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934 (2020) 11
2. Bodla, N., Singh, B., Chellappa, R., Davis, L.S.: Soft-nms–improving object detection with one line of code. In: Proceedings of the IEEE international conference on computer vision. pp. 5561–5569 (2017) 10
3. Cai, Z., Vasconcelos, N.: Cascade r-cnn: Delving into high quality object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 6154–6162 (2018) 1, 3, 4, 5, 11
4. Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., Zhang, Z., Cheng, D., Zhu, C., Cheng, T., Zhao, Q., Li, B., Lu, X., Zhu, R., Wu, Y., Dai, J., Wang, J., Shi, J., Ouyang, W., Loy, C.C., Lin, D.: MMDetection: Open mmlab detection toolbox and benchmark. arXiv preprint arXiv:1906.07155 (2019) 10
5. Chen, Y., Han, C., Wang, N., Zhang, Z.: Revisiting feature alignment for one-stage object detection. arXiv preprint arXiv:1908.01570 (2019) 3, 4, 11, 12
6. Dai, J., Li, Y., He, K., Sun, J.: R-fcn: Object detection via region-based fully convolutional networks. In: Advances in neural information processing systems. pp. 379–387 (2016) 1, 4
7. Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., Wei, Y.: Deformable convolutional networks. In: Proceedings of the IEEE international conference on computer vision. pp. 764–773 (2017) 3, 11
8. Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., Tian, Q.: Centernet: Keypoint triplets for object detection. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 6569–6578 (2019) 2, 3, 5, 6, 7, 10, 11, 12, 13, 14
9. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International journal of computer vision **88**(2), 303–338 (2010) 10
10. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 1440–1448 (2015) 1, 3
11. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 580–587 (2014) 1, 3
12. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 2961–2969 (2017) 4, 8
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016) 1, 3
14. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4700–4708 (2017) 3
15. Huang, L., Yang, Y., Deng, Y., Yu, Y.: Densebox: Unifying landmark localization with end to end object detection. arXiv preprint arXiv:1509.04874 (2015) 3
16. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. Computer science (2014) 10
17. Kong, T., Sun, F., Liu, H., Jiang, Y., Li, L., Shi, J.: Foveabox: Beyound anchor-based object detection. IEEE Transactions on Image Processing (2020) 2, 3, 11
18. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. pp. 1097–1105 (2012) 1

19. Law, H., Deng, J.: Cornernet: Detecting objects as paired keypoints. In: European conference on computer vision. pp. 734–750 (2018) 2, 3, 5, 6, 10, 11, 13, 14
20. Law, H., Teng, Y., Russakovsky, O., Deng, J.: Cornernet-lite: Efficient keypoint based object detection. arXiv preprint arXiv:1904.08900 (2019) 14
21. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015) 1
22. Li, Y., Chen, Y., Wang, N., Zhang, Z.: Scale-aware trident networks for object detection. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 6054–6063 (2019) 4, 11
23. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2117–2125 (2017) 3, 4, 9, 11
24. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. pp. 2980–2988 (2017) 1, 3, 4, 8, 9, 11
25. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision. pp. 740–755 (2014) 3, 9
26. Liu, S., Qi, L., Qin, H., Shi, J., Jia, J.: Path aggregation network for instance segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 8759–8768 (2018) 11, 12
27. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: European conference on computer vision. pp. 21–37 (2016) 1, 3, 4, 5
28. Lu, X., Li, B., Yue, Y., Li, Q., Yan, J.: Grid r-cnn. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7363–7372 (2019) 11
29. Newell, A., Huang, Z., Deng, J.: Associative embedding: End-to-end learning for joint detection and grouping. In: Advances in neural information processing systems. pp. 2277–2287 (2017) 3
30. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: European conference on computer vision. pp. 483–499 (2016) 3, 10
31. Pang, J., Chen, K., Shi, J., Feng, H., Ouyang, W., Lin, D.: Libra r-cnn: Towards balanced learning for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 821–830 (2019) 4, 5, 11
32. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch (2017) 10
33. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 779–788 (2016) 4
34. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems. pp. 91–99 (2015) 1, 2, 3, 4, 5, 14
35. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014) 1
36. Sun, K., Xiao, B., Liu, D., Wang, J.: Deep high-resolution representation learning for human pose estimation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5693–5703 (2019) 3
37. Tan, M., Le, Q.V.: Efficientnet: Rethinking model scaling for convolutional neural networks. arXiv preprint arXiv:1905.11946 (2019) 11

38. Tan, M., Pang, R., Le, Q.V.: Efficientdet: Scalable and efficient object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 10781–10790 (2020) 11
39. Tian, Z., Shen, C., Chen, H., He, T.: Fcos: Fully convolutional one-stage object detection. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 9627–9636 (2019) 2, 3, 5, 11, 14
40. Tychsen-Smith, L., Petersson, L.: Denet: Scalable real-time object detection with directed sparse sampling. In: Proceedings of the IEEE international conference on computer vision. pp. 428–436 (2017) 8
41. Wang, J., Chen, K., Yang, S., Loy, C.C., Lin, D.: Region proposal by guided anchoring. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2965–2974 (2019) 11
42. Yang, Z., Liu, S., Hu, H., Wang, L., Lin, S.: Reppoints: Point set representation for object detection. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 9657–9666 (2019) 2, 11
43. Yu, F., Wang, D., Shelhamer, E., Darrell, T.: Deep layer aggregation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2403–2412 (2018) 3, 10, 14
44. Yu, J., Jiang, Y., Wang, Z., Cao, Z., Huang, T.: Unitbox: An advanced object detection network. In: Proceedings of the 24th ACM international conference on Multimedia. pp. 516–520 (2016) 3
45. Zhang, S., Chi, C., Yao, Y., Lei, Z., Li, S.Z.: 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. pp. 9759–9768 (2020) 11
46. Zhang, S., Wen, L., Bian, X., Lei, Z., Li, S.Z.: Single-shot refinement neural network for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4203–4212 (2018) 1, 3, 4
47. Zhang, X., Wan, F., Liu, C., Ji, R., Ye, Q.: Freeanchor: Learning to match anchors for visual object detection. In: Advances in Neural Information Processing Systems. pp. 147–155 (2019) 3, 11
48. Zhou, X., Wang, D., Krähenbühl, P.: Objects as points. arXiv preprint arXiv:1904.07850 (2019) 3, 10, 11
49. Zhou, X., Zhuo, J., Krahenbuhl, P.: Bottom-up object detection by grouping extreme and center points. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 850–859 (2019) 3, 11
50. Zhu, C., Chen, F., Shen, Z., Savvides, M.: Soft anchor-point object detection. arXiv preprint arXiv:1911.12448 (2019) 2, 3, 11
51. Zhu, C., He, Y., Savvides, M.: Feature selective anchor-free module for single-shot object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 840–849 (2019) 11