CenterNet: Keypoint Triplets for Object Detection

Kaiwen Duan[1]* Song Bai[2] Lingxi Xie[3] Honggang Qi[1,4] Qingming Huang[1,4,5 †] Qi Tian[3 †]

[1]University of Chinese Academy of Sciences
[2]Huazhong University of Science and Technology [3]Huawei Noah's Ark Lab
[4]Key Laboratory of Big Data Mining and Knowledge Management, UCAS [5]Peng Cheng Laboratory
duankaiwen17@mails.ucas.ac.cn {songbai.site,198808xc}@gmail.com
{hgqi,qmhuang}@ucas.ac.cn tian.qi1@huawei.com

Abstract

In object detection, keypoint-based approaches often experience the drawback of a large number of incorrect object bounding boxes, arguably due to the lack of an additional assessment inside cropped regions. This paper presents an efficient solution that explores the visual patterns within individual cropped regions with minimal costs. We build our framework upon a representative one-stage keypoint-based detector named CornerNet. Our approach, named CenterNet, detects each object as a triplet, rather than a pair, of keypoints, which improves both precision and recall. Accordingly, we design two customized modules, cascade corner pooling, and center pooling, that enrich information collected by both the top-left and bottom-right corners and provide more recognizable information from the central regions. On the MS-COCO dataset, CenterNet achieves an AP of ***47.0%****, outperforming all existing one-stage detectors by at least* ***4.9%****. Furthermore, with a faster inference speed than the top-ranked two-stage detectors, CenterNet demonstrates a comparable performance to these detectors. Code is available at* `https://github.com/Duankaiwen/CenterNet`.

1. Introduction

Object detection has been significantly improved with the help of deep learning, especially convolutional neural networks [12] (CNNs). In the current era, one of the most popular flowcharts, the anchor-based flowchart [11, 13, 28, 32, 34], places a set of rectangles with predefined sizes (anchors) on an image and regresses the anchors to the desired place with the help of ground-truth objects. These approaches often require a large number of anchors to ensure a sufficiently high IoU (intersection over union) rate with the ground-truth objects, and the size and aspect ratio of each anchor must be manually set. In addition, anchors and the convolutional features are usually misaligned, which is not conducive to the bounding box classification task.

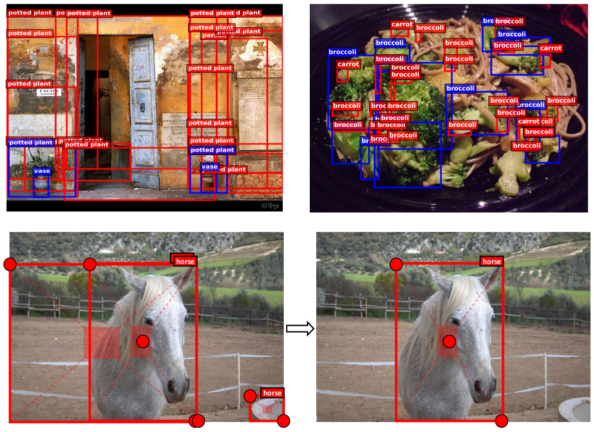


Figure 1: In the first row, we visualize the top 100 bounding boxes (according to the MS-COCO dataset standard) of CornerNet. Ground-truth and predicted objects are marked in blue and red, respectively. In the second row, we show that correct predictions can be determined by checking the central parts of the boxes.

To overcome the drawbacks of anchor-based approaches, a keypoint-based object detection pipeline named CornerNet [21] was proposed. This pipeline represents each object using a pair of corner keypoints, which bypasses the need for anchor boxes and achieves state-of-the-art one-stage object detection accuracy. Nevertheless, the performance of CornerNet is still restricted by its relatively weak ability to refer to the global information of an object. That is, because each object is constructed by a pair of corners, the algorithm sensitively detects the boundaries of objects without being aware of which pairs of keypoints that should be grouped into objects. Consequently, as shown in Figure 1, CornerNet often generates incorrect bounding boxes, most of which could be easily filtered out with some complementary information, *e.g.*, the aspect ratio.

To address this issue, we equip CornerNet with the abil-

*This work was done when the first author was interning at Huawei Noah's Ark Lab.

†Qingming Huang and Qi Tian are the first and second corresponding authors of the paper, respectively.

ity to perceive the visual patterns within each proposed region, enabling it to identify the correctness of each bounding box by itself. In this paper, we present a low-cost yet effective solution named **CenterNet**, which explores the central part of a proposal, *i.e.*, the region that is close to the geometric center of a box, with one extra keypoint. We intuit that if a predicted bounding box has a high IoU with the ground-truth box, then the probability that the center keypoint in the central region of the bounding box will be predicted as the same class is high, and vice versa. Thus, during inference, after a proposal is generated as a pair of corner keypoints, we determine if the proposal is indeed an object by checking if there is a center keypoint of the same class falling within its central region. The idea, as shown in Figure 1, is to use a triplet, instead of a pair, of keypoints to represent each object.

Accordingly, to improve the detection of center keypoints and corners, we propose two strategies to enrich center and corner information, respectively. The first strategy is **center pooling**, which is used in the branch for predicting center keypoints. Center pooling helps the center keypoints obtain more recognizable visual patterns within objects, which makes it easier to perceive the central part of a proposal. We achieve this by obtaining the maximum summed response in both the horizontal and vertical directions of the center keypoint on a feature map for center keypoint prediction. The second strategy is **cascade corner pooling**, which equips the original corner pooling module [21] with the ability to perceive internal information. We achieve this by obtaining the maximum summed response in both the boundary and internal directions of objects on a feature map for corner prediction. Empirically, we verify this two-directional pooling method is more stable, *i.e.*, more robust to feature-level noises, which would contribute to the improvement of both precision and recall.

We evaluate the proposed CenterNet on the MS-COCO dataset [26], one of the most popular benchmarks for large-scale object detection. CenterNet, which incorporates both center pooling and cascade corner pooling, reports an AP of $\mathbf{47.0}\%$ on the test-dev set, outperforming all existing one-stage detectors by a large margin. With an average inference time of $270\mathrm{ms}$ using a 52-layer hourglass backbone [30] per image and $340\mathrm{ms}$ using a 104-layer hourglass backbone [30] per image, CenterNet is quite efficient yet closely matches the state-of-the-art performance of the other two-stage detectors.

2. Related Work

Object detection involves locating and classifying objects. In the deep learning era, powered by deep convolutional neural networks, object detection approaches can be roughly categorized into two main types of pipelines, namely, two-stage approaches and one-stage approaches.

Two-stage approaches divide the object detection task into two stages: extract RoIs (Region of Interesting) and then classify and regress the RoIs.

R-CNN [12] uses a selective search method [45] to locate RoIs in the input images and uses a DCN-based region-wise classifier to classify the RoIs independently. SPP-Net [14] and Fast-RCNN [11] improve R-CNNs by extracting RoIs from the feature maps. Faster-RCNN [34] is allowed to be trained end to end by introducing RPN (region proposal network). RPN can generate RoIs by regressing the anchor boxes. Later, the anchor boxes are widely used in the object detection task. Mask-RCNN [13] adds a mask prediction branch on Faster-RCNN and can thereby detect objects and predict their masks at the same time. R-FCN [6] replaces fully connected layers with position-sensitive score maps to improve the detection of objects. Cascade R-CNN [4] addresses the problem of overfitting at training and quality mismatch at inference by training a sequence of detectors with increasing IoU thresholds. keypoint-based object detection approaches [43, 29, 50, 49] are proposed to avoid the disadvantages of the use of anchor boxes and bounding box regression. Other meaningful works are proposed for different problems in object detection, *e.g.*, a [52, 22] focus on the architecture design, a [1, 10, 37, 47] focus on the contextual relationship, and a [23, 3] focus on the multi-scale unification.

One-stage approaches remove the RoI extraction process and directly classify and regress the candidate anchor boxes.

YOLO [32] uses fewer anchor boxes than other approaches (divide the input image into an $\mathrm{S} \times \mathrm{S}$ grid) to perform regression and classification. YOLOv2 [33] improves the performance by using more anchor boxes and a new bounding box regression method. SSD [28] places anchor boxes densely over an input image and uses features from different convolutional layers to regress and classify the anchor boxes. DSSD [9] introduces a deconvolution module into SSD to combine low- and high-level features. While R-SSD [18] uses pooling and deconvolution operations in different feature layers to combine low-level and high-level features. RON [20] proposes a reverse connection and an objectness prior to extract multiscale features effectively. RefineDet [48] refines the locations and sizes of the anchor boxes twice, exploiting the merits of both one-stage and two-stage approaches. CornerNet [21] is another keypoint-based approach that directly detects an object using a pair of corners. Although CornerNet achieves high performance, it still has room for improvement.

3. Our Approach

3.1. Baseline and Motivation

This paper uses CornerNet [21] as the baseline. For detecting corners, CornerNet produces two heatmaps: a

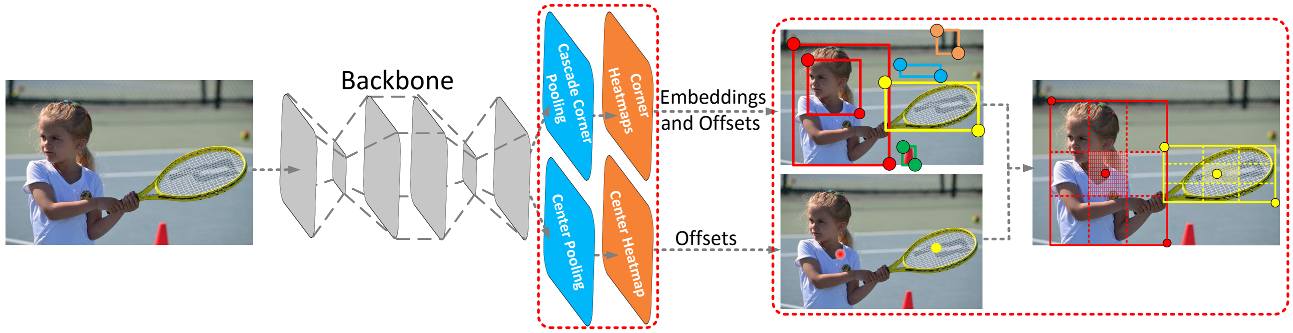


Figure 2: Architecture of CenterNet. A convolutional backbone network applies cascade corner pooling and center pooling to output two corner heatmaps and a center keypoint heatmap, respectively. Similar to CornerNet, a pair of detected corners and the similar embeddings are used to detect a potential bounding box. Then the detected center keypoints are used to determine the final bounding boxes.

Method	FD	FD_5	FD_{25}	FD_{50}	FD_S	FD_M	FD_L
CornerNet	37.8	32.7	36.8	43.8	60.3	33.2	25.1

Table 1: False discovery rates (%) of CornerNet. The false discovery rate reflects the distribution of incorrect bounding boxes. The results suggest that the incorrect bounding boxes account for a large proportion of all bounding boxes.

heatmap of the top-left corners and a heatmap of the bottom-right corners. The heatmaps represent the locations of keypoints of different categories and assign a confidence score to each keypoint. In addition, CornerNet also predicts the embedding and a group of offsets for each corner. The embeddings are used to identify whether two corners are from the same object. The offsets learn to remap the corners from the heatmaps to the input image. To generate object bounding boxes, top-k left-top corners and bottom-right corners are selected from the heatmaps according to their scores. Then, the distance of the embedding vectors of a pair of corners is calculated to determine if the paired corners belong to the same object. An object bounding box is generated if the distance is less than a threshold. The bounding box is assigned a confidence score equal to the average scores of the corner pair.

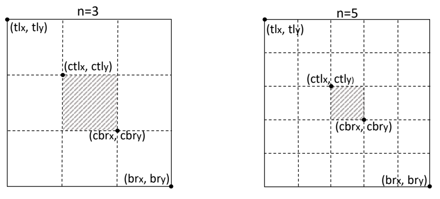
In Table 1, we provide a detailed analysis of CornerNet. We calculate the FD[1] (false discovery) rate of CornerNet on the MS-COCO validation dataset, defined as the proportion of incorrect bounding boxes. The quantitative results demonstrate that the incorrect bounding boxes account for a large proportion of all bounding boxes even at low IoU thresholds, *e.g.*, CornerNet obtains a 32.7% FD rate at IoU $= 0.05$. This means, 32.7 out of every 100 object bounding boxes have an IoU lower than 0.05 with the ground-truth. The FD rate of the small incorrect bounding boxes, with a value of 60.3%, is even higher, than that of larger bounding boxes. One of the possible reasons for this result is that CornerNet cannot assess the regions inside the bounding boxes. One potential method to make CornerNet [21] perceive the visual patterns in bounding boxes is to adapt CornerNet into a two-stage detector, which uses the RoI pooling [11] to assess the visual patterns in bounding boxes. However, such a paradigm is known to be computationally expensive.

In this paper, we propose a highly efficient alternative called **CenterNet** to explore the visual patterns within each bounding box. For object detection, our approach uses a triplet, rather than a pair, of keypoints. By doing so, our approach still keeps a one-stage detector, but partially inherits the functionality of RoI pooling. Our approach only considers the center information, and the cost is minimal. In addition, we further introduce the visual patterns within objects into the keypoint detection process by using center pooling and cascade corner pooling.

3.2. Object Detection as Keypoint Triplets

The overall network architecture is shown in Figure 2. We represent each object using a center keypoint and a pair of corners. Specifically, we embed a heatmap for the center keypoints on the basis of CornerNet and predict the offsets of the center keypoints. Then, we use the method proposed in CornerNet [21] to generate top-k bounding boxes. However, to effectively filter out incorrect bounding boxes, we leverage the detected center keypoints and conduct the following procedure: (1) select top-k center keypoints according to their scores; (2) use the corresponding offsets to remap these center keypoints to the input image; (3) define a central region for each bounding box and check whether the central region contains center keypoints. Note that the class labels of the checked center keypoints should be the same as the class label that of the bounding box; (4) if a center keypoint is detected in the central region, we preserve the bounding box. The score of the bounding box is replaced by the average scores of the triple points, *i.e.*, the top-left

[1] FD $= 1 - \text{AP}$, where AP denotes the average precision at IoU $= [0.05 : 0.05 : 0.5]$ on the MS-COCO dataset. Additionally, $\text{FD}_i = 1 - \text{AP}_i$, where AP_i denotes the average precision at IoU $= i/100$, $\text{FD}_\text{scale} = 1 - \text{AP}_\text{scale}$, where scale $= \{\text{small}, \text{medium}, \text{large}\}$, denotes the scale of the object.



(a) (b)

Figure 3: (a) The central region when $n = 3$. (b) The central region when $n = 5$. The solid rectangles denote the predicted bounding boxes and the shaded regions denote the scalable central regions.

corner, the bottom-right corner, and the center keypoint. If there are no center keypoints detected in the central region, the bounding box will be removed.

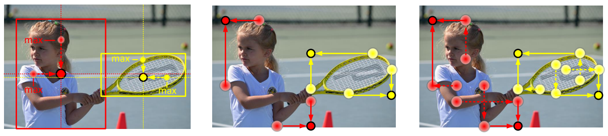
The size of the central region in the bounding box affects the detection results. For example, small central regions lead to a low recall rate for small bounding boxes, while large central regions lead to a low precision for large bounding boxes. Therefore, we propose a scale-aware central region to adaptively fit the size of bounding boxes. The scale-aware central region tends to generate a relatively large central region for a small bounding box and a relatively small central region for a large bounding box. Let $\mathrm{tl_x}$ and $\mathrm{tl_y}$ denote the coordinates of the top-left corner of i and $\mathrm{br_x}$ and $\mathrm{br_y}$ denote the coordinates of the bottom-right corner of i. Define a central region j. Let $\mathrm{ctl_x}$ and $\mathrm{ctl_y}$ denote the coordinates of the top-left corner of j and $\mathrm{cbr_x}$ and $\mathrm{cbr_y}$ denote the coordinates of the bottom-right corner of j. Then $\mathrm{tl_x}$, $\mathrm{tl_y}$, $\mathrm{br_x}$, $\mathrm{br_y}$, $\mathrm{ctl_x}$, $\mathrm{ctl_y}$, $\mathrm{cbr_x}$ and $\mathrm{cbr_y}$ should satisfy the following relationship:

$$\begin{cases} \mathrm{ctl_x} = \dfrac{(n+1)\mathrm{tl_x} + (n-1)\mathrm{br_x}}{2n} \\ \mathrm{ctl_y} = \dfrac{(n+1)\mathrm{tl_y} + (n-1)\mathrm{br_y}}{2n} \\ \mathrm{cbr_x} = \dfrac{(n-1)\mathrm{tl_x} + (n+1)\mathrm{br_x}}{2n} \\ \mathrm{cbr_y} = \dfrac{(n-1)\mathrm{tl_y} + (n+1)\mathrm{br_y}}{2n} \end{cases} \tag{1}$$

where n is odd and determines the scale of the central region j. In this paper, n is set to be 3 and 5 for the scales of bounding boxes less than and greater than 150, respectively. Figure 3 shows two central regions when $n = 3$ and $n = 5$, respectively. According to Equation (1), we can determine a scale-aware central region and then check whether the central region contains center keypoints.

3.3. Enriching Center and Corner Information

Center pooling. The geometric centers of objects do not always convey very recognizable visual patterns (*e.g.*, the human head contains strong visual patterns, but the center keypoint is often in the middle of the human body). To address this issue, we propose center pooling to capture richer and more recognizable visual patterns. Figure 4(a) shows the principle of center pooling. The detailed process of center pooling is as follows: the backbone outputs a feature map and to determine whether a pixel in the feature map is a center keypoint, we need to find the maximum value in both the horizontal and vertical directions and add these values together. By doing so, center pooling helps improve the detection of center keypoints.



(a) (b) (c)

Figure 4: (a) Center pooling takes the maximum values in both horizontal and vertical directions. (b) Corner pooling only takes the maximum values in boundary directions. (c) Cascade corner pooling takes the maximum values in both boundary directions and internal directions of objects.

Cascade corner pooling. Corners are often outside objects, which lack local appearance features. CornerNet [21] uses corner pooling to address this issue. The principle of corner pooling is shown in Figure 4(b). Corner pooling aims to find the maximum values on the boundary directions to determine corners. However, this makes corners sensitive to edges. To address this problem, we need to enable corners to extract features from central regions of the object. The principle of cascade corner pooling is presented in Figure 4(c). Cascade corner pooling first looks along a boundary to find a maximum boundary value and then looks inside the box along with the location of the boundary maximum value[2] to find an internal maximum value; finally, the two maximum values are added together. By cascade corner pooling, the corners obtain both the boundary information and the visual patterns of objects.

Both center pooling and the cascade corner pooling can be easily achieved by applying the corner pooling [21] in different directions. Figure 5(a) shows the structure of the center pooling module. To take a maximum value in a specific direction, *e.g.*, the horizontal direction, we only need to connect the left pooling and the right pooling in sequence. Figure 5(b) shows the structure of a cascade top corner pooling module, in which the white rectangle denotes a 3 × 3 convolution followed by batch normalization. Compared with the top corner pooling in CornerNet [21], a left corner pooling is added before the top corner pooling.

3.4. Training and Inference

Training. Our method is implemented in Pytorch [31] and the network is trained from scratch. The resolution of the

[2] For the topmost, leftmost, bottommost and rightmost boundary, look vertically towards the bottom, horizontally towards the right, vertically towards the top and horizontally towards the left, respectively.

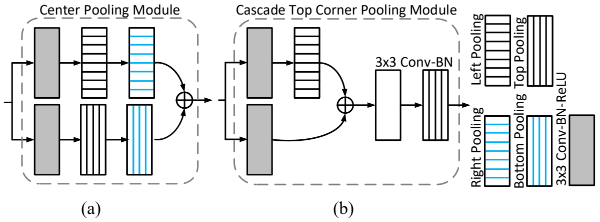


Figure 5: The structures of the center pooling module (a) and the cascade top corner pooling module (b). We achieve center pooling and cascade corner pooling by combining corner pooling in different directions.

input image is 511×511, leading to heatmaps of the size 128×128. We use the data augmentation strategy presented in [21] to train a robust model. Adam [19] is used to optimize the training loss:

$$\mathrm{L} = \mathrm{L}_{\mathrm{det}}^{\mathrm{co}} + \mathrm{L}_{\mathrm{det}}^{\mathrm{ce}} + \alpha \mathrm{L}_{\mathrm{pull}}^{\mathrm{co}} + \beta \mathrm{L}_{\mathrm{push}}^{\mathrm{co}} + \gamma \left(\mathrm{L}_{\mathrm{off}}^{\mathrm{co}} + \mathrm{L}_{\mathrm{off}}^{\mathrm{ce}} \right), \qquad (2)$$

where $\mathrm{L}_{\mathrm{det}}^{\mathrm{co}}$ and $\mathrm{L}_{\mathrm{det}}^{\mathrm{ce}}$ denote the focal losses, which are used to train the network to detect corners and center keypoints, respectively. $\mathrm{L}_{\mathrm{pull}}^{\mathrm{co}}$ is a "pull" loss for corners, which is used to minimize the distance of the embedding vectors that belongs to the same objects. $\mathrm{L}_{\mathrm{push}}^{\mathrm{co}}$ is a "push" loss for corners that is used to maximize the distance of the embedding vectors that belong to different objects. $\mathrm{L}_{\mathrm{off}}^{\mathrm{co}}$ and $\mathrm{L}_{\mathrm{off}}^{\mathrm{ce}}$ are ℓ_1-losses [11], which are used to train the network to predict the offsets of corners and center keypoints, respectively. α, β and γ denote the weights for corresponding losses and are set to 0.1, 0.1 and 1, respectively. $\mathrm{L}_{\mathrm{det}}$, $\mathrm{L}_{\mathrm{pull}}$, $\mathrm{L}_{\mathrm{push}}$ and $\mathrm{L}_{\mathrm{off}}$ are all defined in CornerNet, and we suggest referring to [21] for details. We train CenterNet on 8 Tesla V100 (32GB) GPUs and use a batch size of 48. The maximum number of iterations is $480\mathrm{K}$. We use a learning rate of 2.5×10^{-4} for the first $450\mathrm{K}$ iterations and then continue training $30\mathrm{K}$ iterations with a rate of 2.5×10^{-5}.

Inference. Following [21], for the single-scale testing, we input both the original and horizontally flipped images with the original resolutions into the network. For multi-scale testing, we input both the original and horizontally flipped images with resolutions of $0.6, 1, 1.2, 1.5$ and 1.8. We select top 70 center keypoints, top 70 top-left corners and top 70 bottom-right corners from the heatmaps to detect the bounding boxes. We flip the bounding boxes detected in the horizontally flipped images and mix them into the original bounding boxes. Soft-nms [2] is used to remove the redundant bounding boxes. We finally select the top 100 bounding boxes according to their scores as the final detection results.

4. Experiments

4.1. Dataset, Metrics and Baseline

We evaluate our method on the MS-COCO dataset [26]. This dataset contains 80 categories and more than 1.5 million object instances. A large number of small objects makes it a very challenging dataset. We use the 'trainval35k' set [16] (*i.e.*, $80\mathrm{K}$ training images and $35\mathrm{K}$ validation images) for training and testing the results on the test-dev set. We use another $5\mathrm{K}$ images in the validation set to perform ablation studies and visualization experiments.

The MS-COCO dataset [26] uses AP and AR metrics to characterize the performance of a detector. AP represents the average precision rate, which is computed over ten different IoU thresholds (*i.e.*, $0.5 : 0.05 : 0.95$) and all categories. AR represents the maximum recall rate, which is computed over a fixed number of detections (*i.e.*, 1, 10 and 100) per image and averaged over all categories and the ten different IoU thresholds. Additionally, AP and AR can be used to evaluate the performance under different object scales, including small objects ($\mathrm{area} < 32^2$), medium objects ($32^2 < \mathrm{area} < 96^2$) and large objects ($\mathrm{area} > 96^2$).

Our baseline is CornerNet [21]. We use the stacked hourglass network [30] with 52 and 104 layers as the backbone – the latter has two hourglass modules while the former has only one. All modifications to the hourglass architecture, made by [21], are preserved. In addition, to show that our approach generalizes to other network architectures, we investigate another backbone named HRNet [40, 41], which enjoys the ability to maintain high-resolution representations throughout feature extraction.

4.2. Comparisons with State-of-the-art Detectors

Table 2 shows a comparison with state-of-the-art detectors on the MS-COCO test-dev set. Compared with the baseline CornerNet [21], the proposed CenterNet achieves a remarkable improvement. For example, CenterNet511-52 (which means that the resolution of input images is 511×511 and the backbone is Hourglass-52) reports a single-scale testing AP of 41.6%, an improvement of 3.8% over 37.8%, and a multi-scale testing AP of 43.5%, an improvement of 4.1% over 39.4%, achieved by CornerNet under the same setting. When using the deeper backbone (*i.e.*, Hourglass-104), the AP improvement over CornerNet are 4.4% (from 40.5% to 44.9%) and 4.9% (from 42.1% to 47.0%) under the single-scale and multi-scale testing, respectively. We also replace the backbone with HRNet-W64 [40, 41], which reports APs of 44.0% and 45.6% at single-scale and multi-scale settings, respectively. These results demonstrate the effectiveness of CenterNet.

In addition, the greatest comes from small objects. For instance, CenterNet511-52 improves the AP for small objects by 5.5% (single-scale) and by 6.4% (multi-scale). For the backbone Hourglass-104, the improvements are 6.2% (single-scale) and 8.1% (multi-scale), respectively. This benefit stems from the center information modeled by the center keypoints: the smaller the scale of an incorrect bounding box, the lower the probability that a center key-

Method	Backbone	Train input	Test input	AP	AP$_{50}$	AP$_{75}$	AP$_{S}$	AP$_{M}$	AP$_{L}$	AR$_{1}$	AR$_{10}$	AR$_{100}$	AR$_{S}$	AR$_{M}$	AR$_{L}$
Two-stage:															
DeNet [43]	ResNet-101 [15]	512×512	512×512	33.8	53.4	36.1	12.3	36.1	50.8	29.6	42.6	43.5	19.2	46.9	64.3
CoupleNet [52]	ResNet-101	ori.	ori.	34.4	54.8	37.2	13.4	38.1	50.8	30.0	45.0	46.4	20.7	53.1	68.5
Faster R-CNN by G-RMI [17]	Inception-ResNet-v2 [42]	∼ 1000×600	∼ 1000×600	34.7	55.5	36.7	13.5	38.1	52.0	-	-	-	-	-	-
Faster R-CNN +++ [15]	ResNet-101	∼ 1000×600	∼ 1000×600	34.9	55.7	37.4	15.6	38.7	50.9	-	-	-	-	-	-
Faster R-CNN w/ FPN [24]	ResNet-101	∼ 1000×600	∼ 1000×600	36.2	59.1	39.0	18.2	39.0	48.2	-	-	-	-	-	-
Faster R-CNN w/ TDM [38]	Inception-ResNet-v2	-	-	36.8	57.7	39.2	16.2	39.8	52.1	**31.6**	**49.3**	**51.9**	**28.1**	**56.6**	**71.1**
D-FCN [7]	Aligned-Inception-ResNet	∼ 1000×600	∼ 1000×600	37.5	58.0	-	19.4	40.1	52.5	-	-	-	-	-	-
Regionlets [46]	ResNet-101	∼ 1000×600	∼ 1000×600	39.3	59.8	-	21.7	43.7	50.9	-	-	-	-	-	-
Soft-NMS [2]	Aligned-Inception-ResNet	∼ 1300×800	∼ 1300×800	40.9	62.8	-	23.3	43.6	53.3	-	-	-	-	-	-
Fitness R-CNN [44]	ResNet-101	512×512	1024×1024	41.8	60.9	44.9	21.5	45.0	57.5	-	-	-	-	-	-
Grid R-CNN w/ FPN [29]	ResNeXt-101	∼ 1300×800	∼ 1300×800	43.2	63.0	46.6	25.1	46.5	55.2	-	-	-	-	-	-
D-RFCN + SNIP (multi-scale) [39]	DPN-98 [5]	∼ 2000×1200	∼ 2000×1200	45.7	**67.3**	51.1	29.3	48.8	57.1	-	-	-	-	-	-
PANet (multi-scale) [27]	ResNeXt-101	∼ 1400×840	∼ 1400×840	**47.4**	67.2	**51.8**	**30.1**	**51.7**	**60.0**	-	-	-	-	-	-
One-stage:															
YOLOv2 [33]	DarkNet-19	544×544	544×544	21.6	44.0	19.2	5.0	22.4	35.5	20.7	31.6	33.3	9.8	36.5	54.4
DSOD300 [35]	DS/64-192-48-1	300×300	300×300	29.3	47.3	30.6	9.4	31.5	47.0	27.3	40.7	43.0	16.7	47.1	65.0
GRP-DSOD320 [36]	DS/64-192-48-1	320×320	320×320	30.0	47.9	31.8	10.9	33.6	46.3	28.0	42.1	44.5	18.8	49.1	65.0
SSD513 [28]	ResNet-101	513×513	513×513	31.2	50.4	33.3	10.2	34.5	49.8	28.3	42.1	44.4	17.6	49.2	65.8
DSSD513 [9]	ResNet-101	513×513	513×513	33.2	53.3	35.2	13.0	35.4	51.1	28.9	43.5	46.2	21.8	49.1	66.4
RefineDet512 (single-scale) [48]	ResNet-101	512×512	512×512	36.4	57.5	39.5	16.6	39.9	51.4	-	-	-	-	-	-
CornerNet511 (single-scale) [21]	Hourglass-52	511×511	ori.	37.8	53.7	40.1	17.0	39.0	50.5	33.9	52.3	57.0	35.0	59.3	74.7
RetinaNet800 [25]	ResNet-101	800×800	800×800	39.1	59.1	42.3	21.8	42.7	50.2	-	-	-	-	-	-
CornerNet511 (multi-scale) [21]	Hourglass-52	511×511	≤1.5×	39.4	54.9	42.3	18.9	41.2	52.7	35.0	53.5	57.7	36.1	60.1	75.1
CornerNet511 (single-scale) [21]	Hourglass-104	511×511	ori.	40.5	56.5	43.1	19.4	42.7	53.9	35.3	54.3	59.1	37.4	61.9	76.9
RefineDet512 (multi-scale) [48]	ResNet-101	512×512	≤2.25×	41.8	62.9	45.7	25.6	45.1	54.1						
CornerNet511 (multi-scale) [21]	Hourglass-104	511×511	≤1.5×	42.1	57.8	45.3	20.8	44.8	56.7	36.4	55.7	60.0	38.5	62.7	77.4
CenterNet511 (single-scale)	Hourglass-52	511×511	ori.	41.6	59.4	44.2	22.5	43.1	54.1	34.8	55.7	60.1	38.6	63.3	76.9
CenterNet511 (single-scale)	HRNet-W64 [41]	511×511	ori.	44.0	62.6	47.1	23.0	47.3	57.8	35.4	56.9	61.7	38.3	66.2	79.6
CenterNet511 (single-scale)	Hourglass-104	511×511	ori.	44.9	62.4	48.1	25.6	47.4	57.4	36.1	58.4	63.3	41.3	67.1	80.2
CenterNet511 (multi-scale)	Hourglass-52	511×511	≤1.8×	43.5	61.3	46.7	25.3	45.3	55.0	36.0	57.2	61.3	41.4	64.0	76.3
CenterNet511 (multi-scale)	HRNet-W64	511×511	≤1.8×	46.3	64.7	49.8	26.6	49.6	59.3	36.8	58.6	62.9	42.1	66.9	79.0
CenterNet511 (multi-scale)	Hourglass-104	511×511	≤1.8×	**47.0**	**64.5**	**50.7**	**28.9**	**49.9**	**58.9**	**37.5**	**60.3**	**64.8**	**45.1**	**68.3**	**79.7**

Table 2: Performance comparison (%) with the state-of-the-art methods on the MS-COCO test-dev dataset. CenterNet outperforms all existing one-stage detectors by a large margin and ranks among the top of state-of-the-art two-stage detectors.

point can be detected in the central region. Figure 6(a) and Figure 6(b) show qualitative comparisons that demonstrate the effectiveness of CenterNet in reducing small incorrect bounding boxes.

CenterNet also leads to a large improvement in reducing medium and large incorrect bounding boxes. As Table 2 shows, CenterNet511-104 improves the single-scale testing AP by 4.7% (from 42.7% to 47.4%) and 3.5% (from 53.9% to 57.4%) for medium and large bounding boxes, respectively. Figure 6(c) and Figure 6(d) show qualitative comparisons of the reduction of medium and large incorrect bounding boxes. Notably, the AR is also significantly improved, with the best performance achieved with multi-scale testing. This is because our approach removes many incorrect bounding boxes, which is equivalent to improving the confidence of those bounding boxes with accurate locations but relatively low scores.

Compared to other one-stage approaches, CenterNet511-52 reports a 41.6% single-scale testing AP. This value is already better than the APs of deeper models (*e.g.*, RetinaNet800 [25] and RefineDet [48]). The best performance of CenterNet is an AP **47.0%**, dramatically surpassing all the published one-stage approaches to our best knowledge.

Finally, the performance of CenterNet is also competitive with those of two-stage approaches, *e.g.*, the single-scale testing AP of CenterNet511-52 is comparable to the two-stage approach Fitness R-CNN [44] (41.6% *vs.* 41.8%) and that of CenterNet511-104 is comparable to D-RFCN + SNIP [39] (44.9% *vs.* 45.7%). Nevertheless, two-stage approaches usually use high-resolution input images (*e.g.*, $\sim 1000 \times 600$), which significantly improves the detection accuracy, especially for small objects. The multi-scale testing AP of **47.0%** achieved by CenterNet511-104 closely matches the state-of-the-art AP of 47.4% achieved by the two-stage detector PANet [27]. We present qualitative detection results in Figure 7.

4.3. Incorrect Bounding Box Reduction

The AP [26] metric reflects how many high quality object bounding boxes (usually $\mathrm{IoU} \geqslant 0.5$) a network can predict, but cannot directly reflect how many incorrect object bounding boxes (usually $\mathrm{IoU} \ll 0.5$) a network generates. The FD rate is a suitable metric that reflects the proportion of incorrect bounding boxes. Table 3 shows the FD rates for CornerNet and CenterNet. CornerNet generates many incorrect bounding boxes even at the $\mathrm{IoU} = 0.05$ threshold, *i.e.*, CornerNet511-52 and CornerNet511-104 obtain 35.2% and 32.7% FD rates, respectively. On the other hand, CornerNet generates more small incorrect bounding boxes than medium and large incorrect bounding boxes, with FD values of 62.5% for CornerNet511-52 and 60.3% for CornerNet511-104, respectively. Our CenterNet decreases the FD rates at all criteria by exploring the central regions. Specifically, the FD rates for small bounding boxes decrease the most, with decreases of 9.5% by CenterNet511-52 and 9.6% by CenterNet511-104. This is also the reason why the AP improvement for small objects is more prominent than that for medium and large objects.

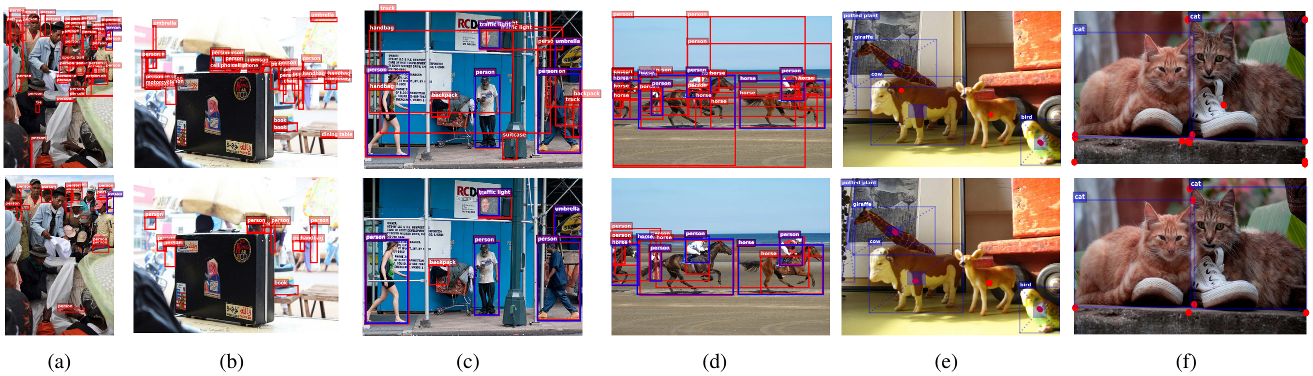


Figure 6: (a) and (b) show that the small incorrect bounding boxes are significantly reduced by the modeling center information. (c) and (d) show that the center information works to reduce the medium and large incorrect bounding boxes. (e) shows the results of the detection of the center keypoints without/with center pooling. (f) shows the results of the detection of corners with corner pooling and cascade corner pooling. The blue boxes above denote the ground-truth. The red boxes and dots denote the predicted bounding boxes and keypoints, respectively.

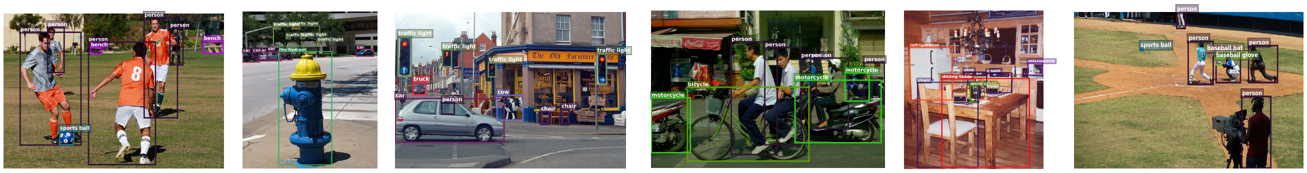


Figure 7: Qualitative detection results on the MS-COCO validation dataset. Only detections with scores higher than 0.5 are shown.

Method	FD	FD_5	FD_{25}	FD_{50}	FD_S	FD_M	FD_L
CornerNet511-52	40.4	35.2	39.4	46.7	62.5	36.9	28.0
CenterNet511-52	**35.1**	**30.7**	**34.2**	**40.8**	**53.0**	**31.3**	**24.4**
CornerNet511-104	37.8	32.7	36.8	43.8	60.3	33.2	25.1
CenterNet511-104	**32.4**	**28.2**	**31.6**	**37.5**	**50.7**	**27.1**	**23.0**

Table 3: Comparison of the false discovery rates (%) of CornerNet and CenterNet on the MS-COCO validation dataset. The results suggest that CenterNet avoids a large number of incorrect bounding boxes, especially for small incorrect bounding boxes.

4.4. Inference Speed

The proposed CenterNet explores the visual patterns within each proposed region with minimal costs. To ensure a fair comparison, we test the inference speed of both CornerNet [21] and CenterNet on an NVIDIA Tesla P100 GPU. We determine that the average inference time of CornerNet511-104 is 300ms per image and that of CenterNet511-104 is 340ms per image. Meanwhile, using the Hourglass-52 backbone can speed up the inference speed. Our CenterNet511-52 takes an average of 270ms to process each image, which is faster and more accurate than CornerNet511-104.

4.5. Ablation Study

Our work contributes three components to object detection, namely, central region exploration, center pooling, and cascade corner pooling. To analyze the contribution of each individual component, an ablation study is given here. The baseline is CornerNet511-52 [21]. We add the three components to the baseline one by one and follow the default parameter setting detailed in Section 4.1. The results are given in Table 4.

Central region exploration. To understand the importance of central region exploration (see CRE in the table), we add a center heatmap branch to the baseline and use a triplet of keypoints to detect the bounding boxes. For center keypoint detection, we only use conventional convolutions. As presented in the third row in Table 4, we improve the AP by 2.3% (from 37.6% to 39.9%). However, we find that the improvement for small objects (that is 4.6%) is more significant than that for other object scales. The improvement for large objects is almost negligible (from 52.2% to 52.3%). This is not surprising, because compared to large objects, small objects are more likely to benefit from being filtered by center keypoints.

Center pooling. To demonstrate the effectiveness of the proposed center pooling, we add the center pooling module to the network (see CTP in the table). The fourth row in Table 4 shows that center pooling improves the AP by 0.9% (from 39.9% to 40.8%). Notably, with the help of center pooling, we increase the AP for large objects by 1.4% (from 52.2% to 53.6%), which is a much higher improvement than that of conventional convolutions (*i.e.*, 1.4% *vs.* 0.1%). this demonstrates that our center pooling is effective in detect-

CRE	CTP	CCP	AP	AP_{50}	AP_{75}	AP_{S}	AP_{M}	AP_{L}	AR_1	AR_{10}	AR_{100}	AR_{S}	AR_{M}	AR_{L}
			37.6	53.3	40.0	18.5	39.6	52.2	33.7	52.2	56.7	37.2	60.0	74.0
		✓	38.3	54.2	40.5	18.6	40.5	52.2	34.0	53.0	57.9	36.6	60.8	75.8
✓			39.9	57.7	42.3	23.1	42.3	52.3	33.8	54.2	58.5	38.7	62.4	74.4
✓	✓		40.8	58.6	43.6	23.6	43.6	53.6	33.9	54.5	59.0	39.0	63.2	74.7
✓	✓	✓	**41.3**	**59.2**	**43.9**	**23.6**	**43.8**	**55.8**	**34.5**	**55.0**	**59.2**	**39.1**	**63.5**	**75.1**

Table 4: Ablation study on the major components of CenterNet511-52 on the MS-COCO validation dataset. The CRE denotes central region exploration, the CTP denotes center pooling, and the CCP denotes cascade corner pooling.

Method	AP	AP_{50}	AP_{75}	AP_{S}	AP_{M}	AP_{L}
CenterNet511-52 w/o GT	41.3	59.2	43.9	23.6	43.8	55.8
CenterNet511-52 w/ GT	**56.5**	**78.3**	**61.4**	**39.1**	**60.3**	**70.3**
CenterNet511-104 w/o GT	44.8	62.4	48.2	25.9	48.9	58.8
CenterNet511-104 w/ GT	**58.1**	**78.4**	**63.9**	**40.4**	**63.0**	**72.1**

Table 5: Error analysis of center keypoints using ground-truth values. We replace the predicted center keypoints with the ground-truth, and the results suggest that there is still room for improvement in the detection of center keypoints.

ing the center keypoints of objects, especially for large objects. Our explanation for this result is that center pooling can extract rich internal visual patterns, and larger objects contain more accessible internal visual patterns than smaller objects. Figure 6(e) shows the results of the detection of center keypoints without/with center pooling. We can see that conventional convolution fails to locate the center keypoint of the cow, but with center pooling, the center keypoint is successfully located.

Cascade corner pooling. We replace corner pooling [21] with cascade corner pooling to detect corners (see CCP in the table). The second row in Table 4 shows the results of the test on the basis of CornerNet511-52. We find that cascade corner pooling improves the AP by 0.7% (from 37.6% to 38.3%). The last row shows the results of the test on the basis of CenterNet511-52, which improves the AP by 0.5% (from 40.8% to 41.3%). The results of the second row show that there is almost no change in the AP for large objects (*i.e.*, 52.2% *vs.* 52.2%), but the AR is improved by 1.8% (from 74.0% to 75.8%). This suggests that cascade corner pooling can detect an increased number of objects due to the rich internal visual patterns of large objects, but visual patterns that are too rich may interfere with the perception of the boundary information, leading to many inaccurate bounding boxes. After equipping with our CenterNet, the inaccurate bounding boxes are effectively suppressed, which improves the AP for large objects by 2.2% (from 53.6% to 55.8%). Figure 6(f) shows the results of corner detection with corner pooling or cascade corner pooling. As shown in the figure, cascade corner pooling successfully locates a pair of corners on the left side of the cat image, while corner pooling fails to locate these corners.

4.6. Error Analysis

The exploration of visual patterns within each bounding box depends on the center keypoints. In other words, once a center keypoint is missed, the proposed CenterNet misses the visual patterns within the bounding box. To understand the importance of center keypoints, we replace the predicted center keypoints with the ground-truth values and evaluate the performance on the MS-COCO validation dataset. Table 5 shows that using the ground-truth center keypoints improves the AP from 41.3% to 56.5% for CenterNet511-52 and from 44.8% to 58.1% for CenterNet511-104. The APs for small, medium and large objects are improved by 15.5%, 16.5%, and 14.5% for CenterNet511-52 and 14.5%, 14.1%, and 13.3% for CenterNet511-104, respectively. Since using ground-truth center keypoints largely boosts detection AP, we believe there is still room for improvement by detecting center keypoints more accurately.

We also investigated the PASCAL VOC 2007 dataset [8]. CenterNet-52 reports single-scale and multi-scale testing APs of 81.9% and 83.6%, and the numbers of CenterNet-104 are 83.3% and 84.5%, respectively. All these results outperform the corresponding models of CornerNet-52 (77.5%, 80.1%) and CornerNet-104 (80.7% and 81.6%).

5. Conclusions

In this paper, we propose CenterNet, which detects objects using a triplet, including one center keypoint and two corners. Our approach addresses the inability of CornerNet to assess the inside of cropped regions by exploring the visual patterns within each proposed region with minimal costs. This is a common defect of all one-stage approaches. As one-stage approaches remove the RoI extraction process, they cannot consider the internal information within cropped regions.

An intuitive explanation of our contribution is that we equip a one-stage detector with the ability of two-stage approaches, with the addition of an efficient discriminator. We believe that our idea of adding an extra branch for the center keypoint can potentially be generalized to other existing one-stage approaches (*e.g.*, SSD [28]). Furthermore, some advanced training strategies [51] can be used to improve performance. This objective will be the subject of our future work.

Acknowledgments This research was supported by National Natural Science Foundation of China: 61620106009, U1636214, 61836002 and 61472388, and Key Research Program of Frontier Sciences, CAS: QYZDJ-SSWSYS013.

References

[1] Sean Bell, C. Lawrence Zitnick, Kavita Bala, and Ross Girshick. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2874–2883, 2016.

[2] Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S Davis. Soft-nms–improving object detection with one line of code. In *Proceedings of the IEEE international conference on computer vision*, pages 5561–5569, 2017.

[3] Zhaowei Cai, Quanfu Fan, Rogerio S Feris, and Nuno Vasconcelos. A unified multi-scale deep convolutional neural network for fast object detection. In *European conference on computer vision*, pages 354–370. Springer, 2016.

[4] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6154–6162, 2018.

[5] Yunpeng Chen, Jianan Li, Huaxin Xiao, Xiaojie Jin, Shuicheng Yan, and Jiashi Feng. Dual path networks. In *Advances in neural information processing systems*, pages 4467–4475, 2017.

[6] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*, pages 379–387, 2016.

[7] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017.

[8] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *international journal of computer vision*, 88(2):303–338, 2010.

[9] Cheng-Yang Fu, Wei Liu, Ananth Ranga, Ambrish Tyagi, and Alexander C Berg. Dssd: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*, 2017.

[10] Spyros Gidaris and Nikos Komodakis. Object detection via a multi-region and semantic segmentation-aware cnn model. In *Proceedings of the IEEE international conference on computer vision*, pages 1134–1142, 2015.

[11] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[12] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1904–1916, 2015.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[16] Derek Hoiem, Yodsawalai Chodpathumwan, and Qieyun Dai. Diagnosing error in object detectors. In *European conference on computer vision*, pages 340–353. Springer, 2012.

[17] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7310–7311, 2017.

[18] Jisoo Jeong, Hyojin Park, and Nojun Kwak. Enhancement of ssd by concatenating feature maps for object detection. *arXiv preprint arXiv:1705.09587*, 2017.

[19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *Computer science*, 2014.

[20] Tao Kong, Fuchun Sun, Anbang Yao, Huaping Liu, Ming Lu, and Yurong Chen. Ron: Reverse connection with objectness prior networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5936–5944, 2017.

[21] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European conference on computer vision*, pages 734–750, 2018.

[22] Hyungtae Lee, Sungmin Eum, and Heesung Kwon. Me r-cnn: Multi-expert r-cnn for object detection. *arXiv preprint arXiv:1704.01069*, 2017.

[23] Yanghao Li, Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Scale-aware trident networks for object detection. *arXiv preprint arXiv:1901.01892*, 2019.

[24] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[25] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[26] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[27] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8759–8768, 2018.

[28] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[29] Xin Lu, Buyu Li, Yuxin Yue, Quanquan Li, and Junjie Yan. Grid r-cnn. 2018.

[30] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, pages 483–499. Springer, 2016.

[31] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[32] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[33] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

[34] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[35] Zhiqiang Shen, Zhuang Liu, Jianguo Li, Yu-Gang Jiang, Yurong Chen, and Xiangyang Xue. Dsod: Learning deeply supervised object detectors from scratch. In *Proceedings of the IEEE international conference on computer vision*, pages 1919–1927, 2017.

[36] Zhiqiang Shen, Honghui Shi, Rogerio Feris, Liangliang Cao, Shuicheng Yan, Ding Liu, Xinchao Wang, Xiangyang Xue, and Thomas S Huang. Learning object detectors from scratch with gated recurrent feature pyramids. *arXiv preprint arXiv:1712.00886*, 2017.

[37] Abhinav Shrivastava and Abhinav Gupta. Contextual priming and feedback for faster r-cnn. In *European conference on computer vision*, pages 330–348, 2016.

[38] Abhinav Shrivastava, Rahul Sukthankar, Jitendra Malik, and Abhinav Gupta. Beyond skip connections: Top-down modulation for object detection. *arXiv preprint arXiv:1612.06851*, 2016.

[39] Bharat Singh and Larry S Davis. An analysis of scale invariance in object detection snip. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3578–3587, 2018.

[40] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. *CoRR*, abs/1902.09212, 2019.

[41] Ke Sun, Yang Zhao, Borui Jiang, Tianheng Cheng, Bin Xiao, Dong Liu, Yadong Mu, Xinggang Wang, Wenyu Liu, and Jingdong Wang. High-resolution representations for labeling pixels and regions. *CoRR*, abs/1904.04514.

[42] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-First AAAI conference on artificial intelligence*, 2017.

[43] Lachlan Tychsen-Smith and Lars Petersson. Denet: Scalable real-time object detection with directed sparse sampling. In *Proceedings of the IEEE international conference on computer vision*, pages 428–436, 2017.

[44] Lachlan Tychsen-Smith and Lars Petersson. Improving object localization with fitness nms and bounded iou loss. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6877–6885, 2018.

[45] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

[46] Hongyu Xu, Xutao Lv, Xiaoyu Wang, Zhou Ren, Navaneeth Bodla, and Rama Chellappa. Deep regionlets for object detection. In *Proceedings of the European conference on computer vision*, pages 798–814, 2018.

[47] Xingyu Zeng, Wanli Ouyang, Bin Yang, Junjie Yan, and Xiaogang Wang. Gated bi-directional cnn for object detection. In *European conference on computer vision*, pages 354–369. Springer, 2016.

[48] Shifeng Zhang, Longyin Wen, Xiao Bian, Zhen Lei, and Stan Z Li. Single-shot refinement neural network for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4203–4212, 2018.

[49] Xingyi Zhou, Dequan Wang, and Philipp Krhenbhl. Objects as points. *arXiv preprint arXiv: 1904.07850*, 2019.

[50] Xingyi Zhou, Jiacheng Zhuo, and Philipp Krahenbuhl. Bottom-up object detection by grouping extreme and center points. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 850–859, 2019.

[51] Rui Zhu, Shifeng Zhang, Xiaobo Wang, Longyin Wen, Hailin Shi, Liefeng Bo, and Tao Mei. Scratchdet: Training single-shot object detectors from scratch. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2019.

[52] Yousong Zhu, Chaoyang Zhao, Jinqiao Wang, Xu Zhao, Yi Wu, and Hanqing Lu. Couplenet: Coupling global structure with local parts for object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 4126–4134, 2017.