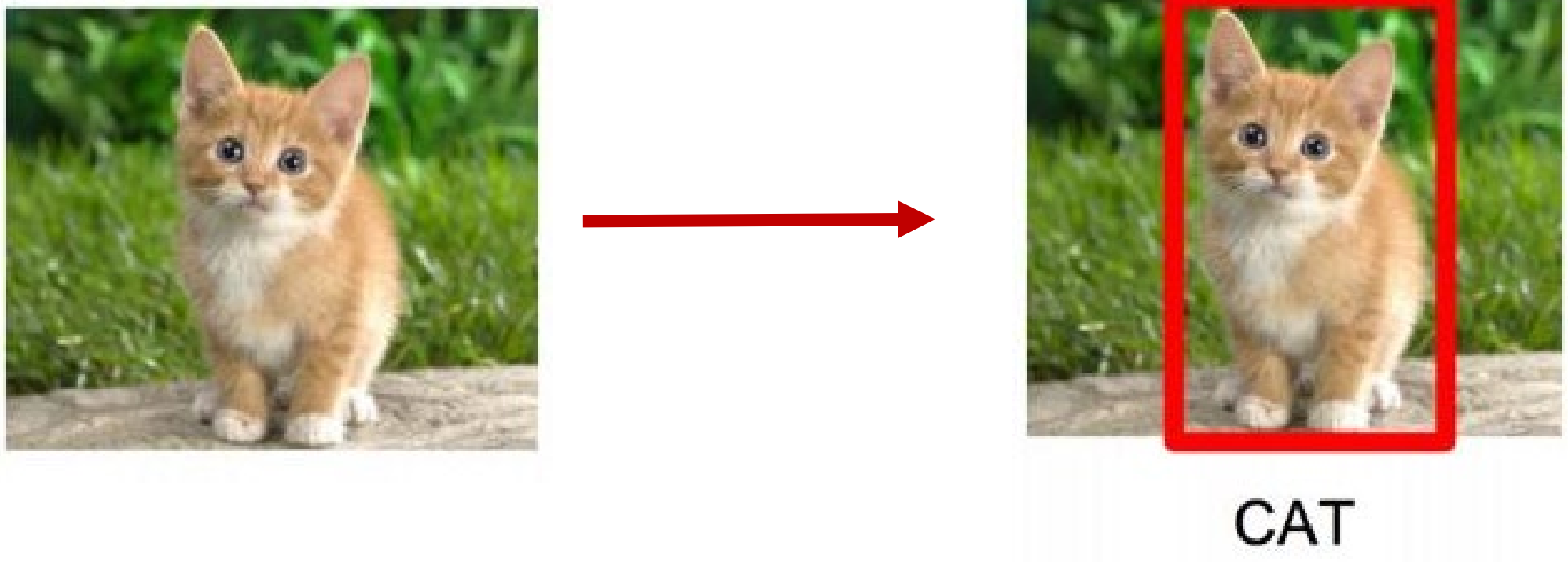
CornerNet: Detecting Objects as Paired Key points

2019.10.23

Introduction

Q: What is object detection ?

A: what & where

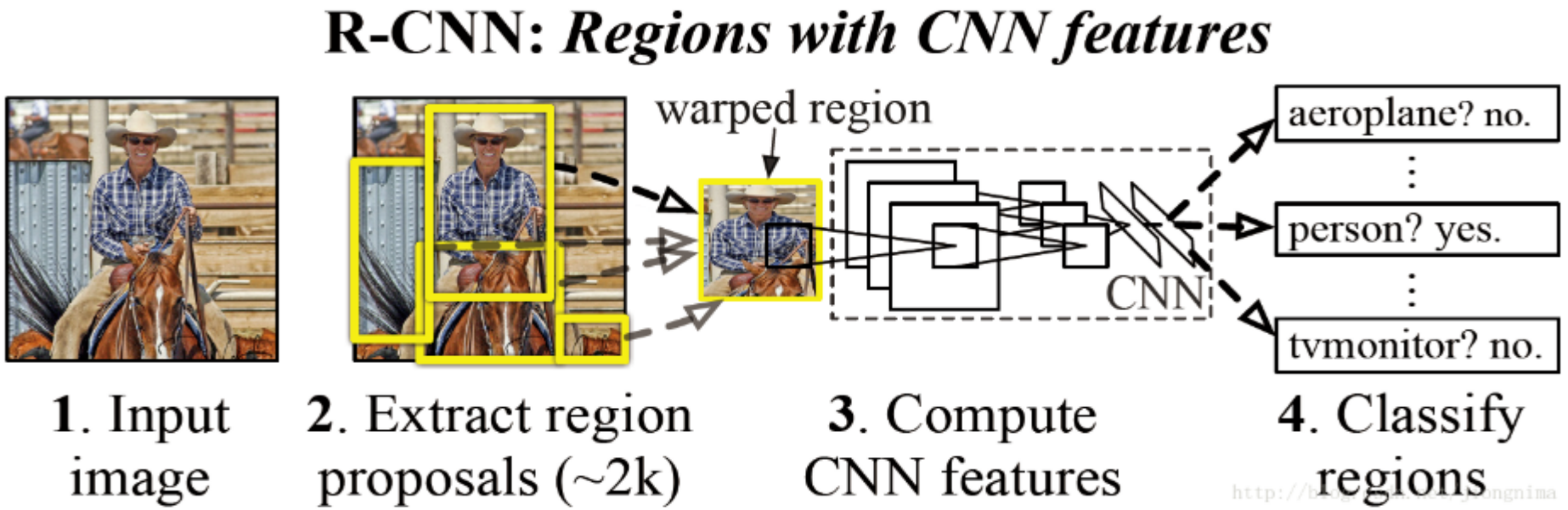


Introduction

Two major object detection algorithms

1.Two stage

Region proposals + Bounding box Regression & object classification



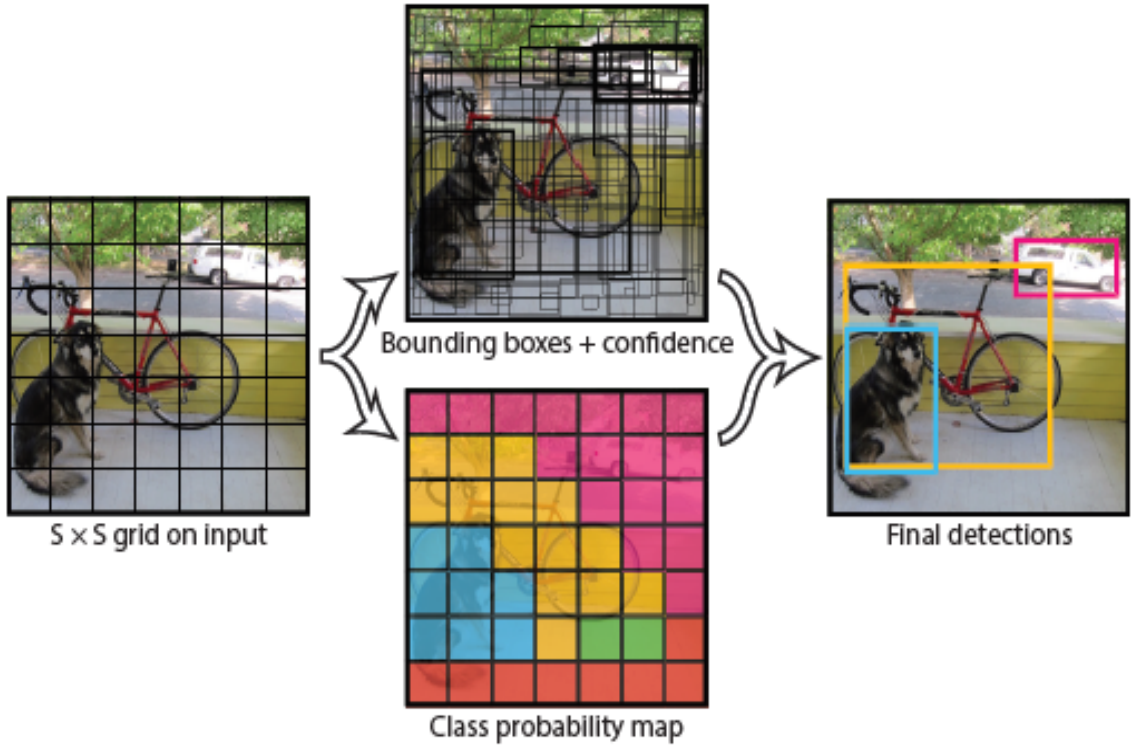
Introduction

Two major object detection algorithms

2.one stage

Got rid of region proposals

A common component of state-of-the-art approaches is anchor boxes, which are boxes of various sizes and aspect ratios that serve as detection candidates. Anchor boxes are extensively used in one-stage detectors, which can achieve results highly competitive with two-stage detectors while being more efcient.



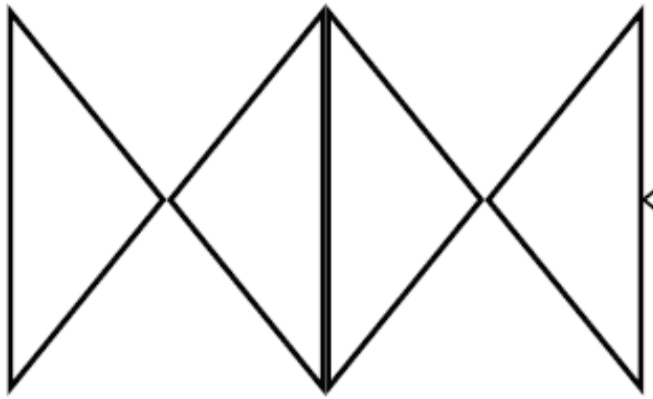
Motivation

The use of anchor boxes has two drawbacks

1.we typically need a very large set of anchor boxes.

2.the use of anchor boxes introduces many hyper parameters and design choices.

Overview



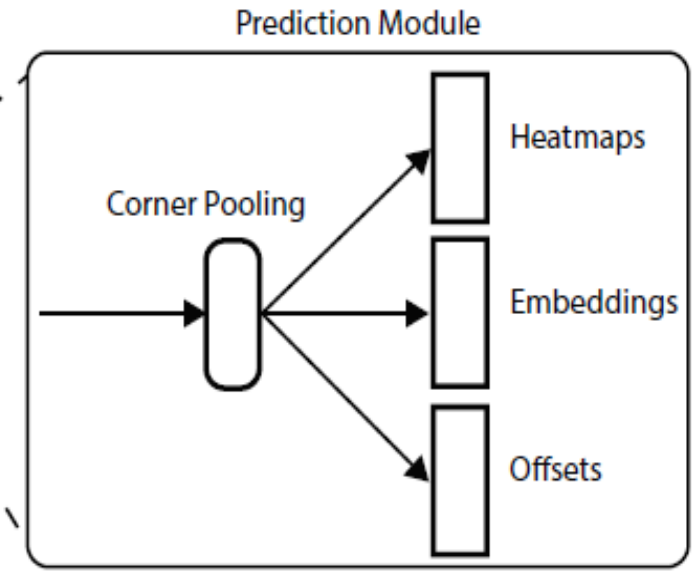
Hourglass Network

Top-left Corners

Prediction Module

Prediction Module

Bottom-right corners



Key technology

Heatmaps

Each set of heatmaps has C channels, where C is the number of categories,and is of size H W.

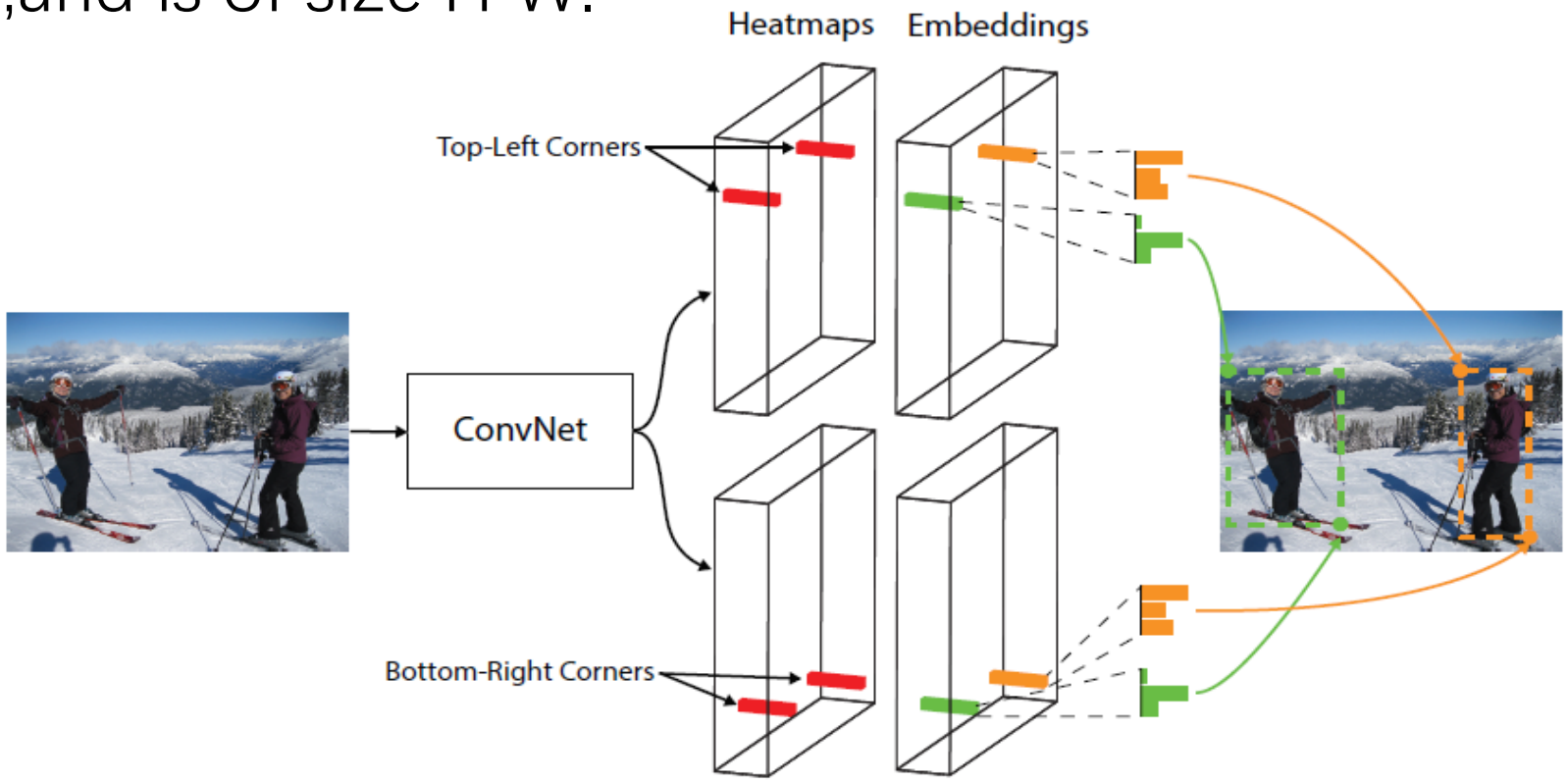


Fig. 1 We detect an object as a pair of bounding box corners grouped together. A convolutional network outputs a heatmap for all top-left corners, a heatmap for all bottom-right corners, and an embedding vector for each detected corner. The network is trained to predict similar embeddings for corners that belong to the same object.

Key technology

Heatmaps

we reduce the penalty given to negative locations within a radius of the positive location.
This is because a pair of false corner detections, if they are close to their respective ground truth locations, can still produce a box that suciently overlaps the ground-truth box

Penalty reduction is given by an unnormalized 2D Gaussian, whose center is at the positive location.

$$e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Fig. 5 "Ground-truth" heatmaps for training. Boxes *(green dotted rectangles)* whose corners are within the radii of the positive locations *(orange circles)* still have large overlaps with the ground-truth annotations *(red solid rectangles)*.

Key technology

Heatmaps

$$L_{det} = \frac{-1}{N} \sum_{c=1}^{C} \sum_{i=1}^{H} \sum_{j=1}^{W} \begin{cases} (1 - p_{cij})^{\alpha} \log(p_{cij}) & \text{if } y_{cij} = 1 \\ (1 - y_{cij})^{\beta} (p_{cij})^{\alpha} \log(1 - p_{cij}) & \text{otherwise} \end{cases}$$

Let p(cij)be the score at location (i,j) for class c in the predicted heatmaps, and let ycij be the ground-truth" heatmap augmented with the unnormalized Gaus-Sians.

where N is the number of objects in an image, and beta are the hyper-parameters which control the contribution of each point, the (1-ycij) term reduces the penalty around the ground truth locations.

Key technology

offsets

Networks use down sampling layers to gather global information. When they are applied to an image fully convolutionally, the size of the output is usually smaller than the image. When we remap the locations from the heatmaps to the input image, some precision may be lost.

$$(x, y) \rightarrow \left(\left\lfloor \frac{x}{n} \right\rfloor, \left\lfloor \frac{y}{n} \right\rfloor\right)$$

n is the downsampling factor

To address this issue we predict location setoffs

$$\boldsymbol{o}_k = \left(\frac{x_k}{n} - \left\lfloor \frac{x_k}{n} \right\rfloor, \frac{y_k}{n} - \left\lfloor \frac{y_k}{n} \right\rfloor\right)$$

$$L_{off} = \frac{1}{N} \sum_{k=1}^{N} \text{SmoothL1Loss}\left(\boldsymbol{o}_k, \hat{\boldsymbol{o}}_k\right)$$

Key technology

Embedding

we use the “pull" loss to train the network to group the corners and the “push“ loss to separate the corners

$$L_{pull} = \frac{1}{N}\sum_{k=1}^{N}\left[(e_{t_k} - e_k)^2 + (e_{b_k} - e_k)^2\right],$$

$$L_{push} = \frac{1}{N(N-1)}\sum_{k=1}^{N}\sum_{\substack{j=1 \\ j\neq k}}^{N}\max\left(0, \Delta - |e_k - e_j|\right),$$

Let etk be the embedding for the top-left corner of object k and ebk for the bottom-right corner. ek is the average of etk and ebk.

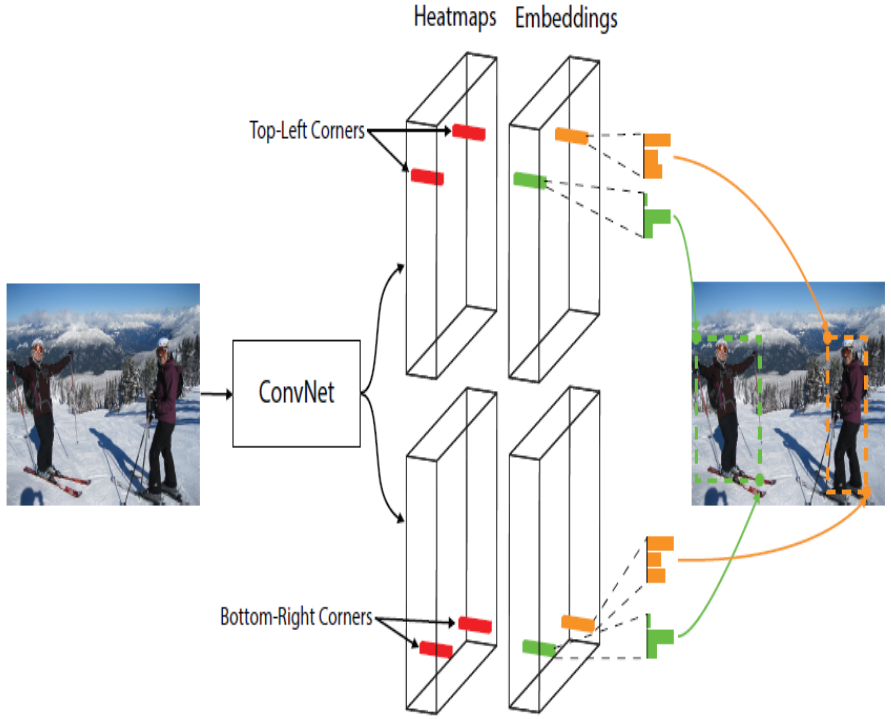


Fig. 1 We detect an object as a pair of bounding box corners grouped together. A convolutional network outputs a heatmap for all top-left corners, a heatmap for all bottom-right corners, and an embedding vector for each detected corner. The network is trained to predict similar embeddings for corners that belong to the same object.

Key technology

Corner pooling

there is often no local visual evidence for the presence of corners.

To determine if a pixel is a top-left corner, we need to look horizontally towards the right for the topmost boundary of an object and vertically towards the bottom for the left most boundary

Fig. 2 Often there is no local evidence to determine the location of a bounding box corner. We address this issue by proposing a new type of pooling layer.

Key technology

Corner pooling

we introduce corner pooling, a new type of pooling layer that helps the network better localize corners.

Suppose we want to determine if a pixel at location (i; j) is a top-left corner.

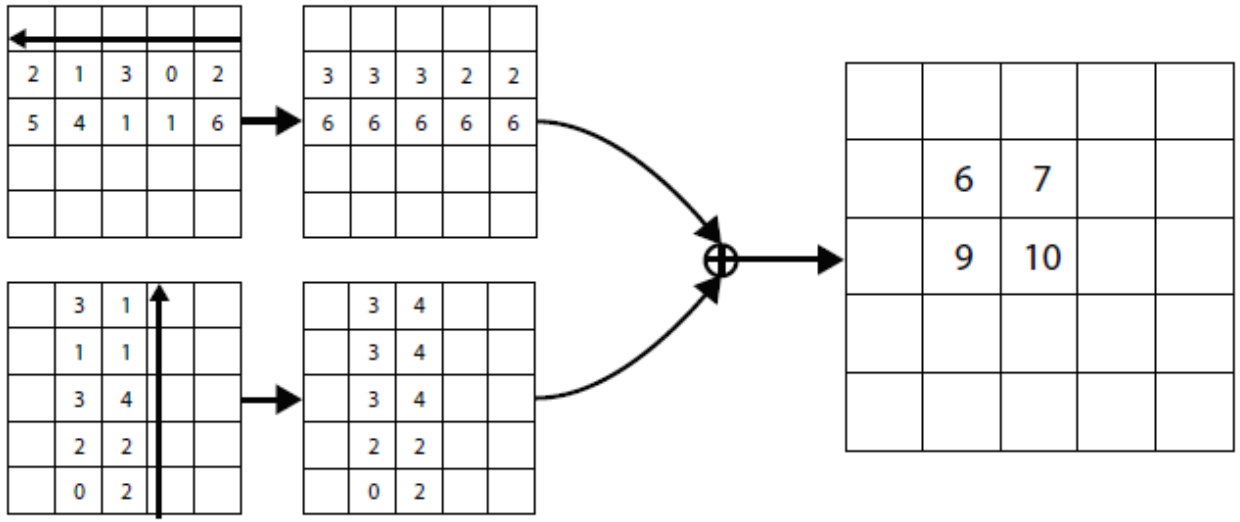


Fig. 6 The top-left corner pooling layer can be implemented very efficiently. We scan from right to left for the horizontal max-pooling and from bottom to top for the vertical max-pooling. We then add two max-pooled feature maps.

Experiment

Table 1 Ablation on corner pooling on MS COCO validation.

	AP	AP^{50}	AP^{75}	AP^s	AP^m	AP^l
w/o corner pooling	36.5	52.0	38.8	17.5	38.9	49.4
w/ corner pooling	38.4	53.8	40.9	18.6	40.5	51.8
improvement	+2.0	+2.1	+2.1	+1.1	+2.4	+3.6

Table 2 Reducing the penalty given to the negative locations near positive locations helps significantly improve the performance of the network

	AP	AP^{50}	AP^{75}	AP^s	AP^m	AP^l
w/o reducing penalty	32.9	49.1	34.8	19.0	37.0	40.7
fixed radius	35.6	52.5	37.7	18.7	38.5	46.0
object-dependent radius	38.4	53.8	40.9	18.6	40.5	51.8

Experiment

Table 7 CornerNet versus others on MS COCO test-dev. CornerNet outperforms all one-stage detectors and achieves results competitive to two-stage detectors

Method	Backbone	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 detectors													
DeNet (Tychsen-Smith and Petersson, 2017a)	ResNet-101	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 (Zhu et al., 2017)	ResNet-101	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 (Huang et al., 2017)	Inception-ResNet-v2 (Szegedy et al., 2017)	34.7	55.5	36.7	13.5	38.1	52.0	-	-	-	-	-	-
Faster R-CNN+++ (He et al., 2016)	ResNet-101	34.9	55.7	37.4	15.6	38.7	50.9	-	-	-	-	-	-
Faster R-CNN w/ FPN (Lin et al., 2016)	ResNet-101	36.2	59.1	39.0	18.2	39.0	48.2	-	-	-	-	-	-
Faster R-CNN w/ TDM (Shrivastava et al., 2016)	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 (Dai et al., 2017)	Aligned-Inception-ResNet	37.5	58.0	-	19.4	40.1	52.5	-	-	-	-	-	-
Regionlets (Xu et al., 2017)	ResNet-101	39.3	59.8	-	21.7	43.7	50.9	-	-	-	-	-	-
Mask R-CNN (He et al., 2017)	ResNeXt-101	39.8	62.3	43.4	22.1	43.2	51.2	-	-	-	-	-	-
Soft-NMS (Bodla et al., 2017)	Aligned-Inception-ResNet	40.9	62.8	-	23.3	43.6	53.3	-	-	-	-	-	-
LH R-CNN (Li et al., 2017)	ResNet-101	41.5	-	-	25.2	45.3	53.1	-	-	-	-	-	-
Fitness-NMS (Tychsen-Smith and Petersson, 2017b)	ResNet-101	41.8	60.9	44.9	21.5	45.0	57.5	-	-	-	-	-	-
Cascade R-CNN (Cai and Vasconcelos, 2017)	ResNet-101	42.8	62.1	46.3	23.7	45.5	55.2	-	-	-	-	-	-
D-RFCN + SNIP (Singh and Davis, 2017)	DPN-98 (Chen et al., 2017)	45.7	67.3	51.1	29.3	48.8	57.1	-	-	-	-	-	-
One-stage detectors													
YOLOv2 (Redmon and Farhadi, 2016)	DarkNet-19	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 (Shen et al., 2017a)	DS/64-192-48-1	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 (Shen et al., 2017b)	DS/64-192-48-1	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 (Liu et al., 2016)	ResNet-101	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 (Fu et al., 2017)	ResNet-101	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) (Zhang et al., 2017)	ResNet-101	36.4	57.5	39.5	16.6	39.9	51.4	-	-	-	-	-	-
RetinaNet800 (Lin et al., 2017)	ResNet-101	39.1	59.1	42.3	21.8	42.7	50.2	-	-	-	-	-	-
RefineDet512 (multi scale) (Zhang et al., 2017)	ResNet-101	41.8	62.9	45.7	25.6	45.1	54.1	-	-	-	-	-	-
CornerNet511 (single scale)	Hourglass-104	40.6	56.4	43.2	19.1	42.8	54.3	35.3	54.7	59.4	37.4	62.4	77.2
CornerNet511 (multi scale)	Hourglass-104	42.2	57.8	45.2	20.7	44.8	56.6	36.6	55.9	60.3	39.5	63.2	77.3

Thanks