FoveaBox: Beyond Anchor-based Object Detector

Tao Kong[1] Fuchun Sun[1] Huaping Liu[1] Yuning Jiang[2] Jianbo Shi[3]
[1]Department of Computer Science and Technology, Tsinghua University,
Beijing National Research Center for Information Science and Technology (BNRist)
[2]ByteDance AI Lab [3]University of Pennsylvania
taokongcn@gmail.com, {fcsun,hpliu}@tsinghua.edu.cn,
jiangyuning@bytedance.com, jshi@seas.upenn.edu

Abstract

We present FoveaBox, an accurate, flexible and completely anchor-free framework for object detection. While almost all state-of-the-art object detectors utilize the predefined anchors to enumerate possible locations, scales and aspect ratios for the search of the objects, their performance and generalization ability are also limited to the design of anchors. Instead, FoveaBox directly learns the object existing possibility and the bounding box coordinates without anchor reference. This is achieved by: (a) predicting category-sensitive semantic maps for the object existing possibility, and (b) producing category-agnostic bounding box for each position that potentially contains an object. The scales of target boxes are naturally associated with feature pyramid representations for each input image.

Without bells and whistles, FoveaBox achieves state-of-the-art single model performance of 42.1 AP on the standard COCO detection benchmark. Specially for the objects with arbitrary aspect ratios, FoveaBox brings in significant improvement compared to the anchor-based detectors. More surprisingly, when it is challenged by the stretched testing images, FoveaBox shows great robustness and generalization ability to the changed distribution of bounding box shapes. The code will be made publicly available.

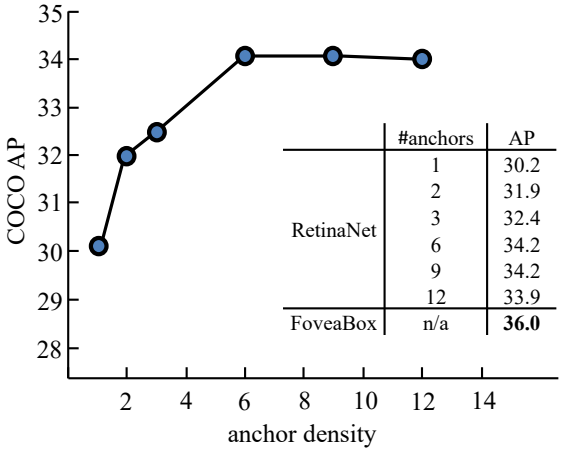


Figure 1. The RetinaNet object detection performances with different anchor numbers $v.s.$ the proposed FoveaNet. With anchor density increasing, the performance gets saturated in RetinaNet. While FoveaNet shows better performance without the utilization of anchors. An improved variant of FoveaNet achieves 42.1 AP with ResNeXt-101 backbone, which is not shown in this figure. More details are given in §4.1.

1. Introduction

Object detection requires the solution of two main tasks: *recognition* and *localization*. Given an arbitrary image, an object detection system needs to determine whether there are any instances of semantic objects from predefined categories and, if present, to return the spatial location and extent. To add the localization functionality to generic object detection systems, sliding window approaches have been the method of choice for many years [25][7].

Recently, deep learning techniques have emerged as powerful methods for learning feature representations automatically from data [43][16]. R-CNN [12] and Fast R-CNN [11] use a few thousands of category-independent region proposals to reduce the searching space for an image. The region proposal generation stage is subsequently replaced by anchor-based Region Proposal Networks (RPN) [40]. Since then, the anchor boxes are widely used as a common component for searching possible regions of interest for modern object detection frameworks. In short, anchor method suggests dividing the box space (including position, scale, aspect ratio) into discrete bins and refining the object box in the corresponding bin. Most state-of-the-art detectors rely on anchor boxes to enumerate the possible locations, scales, and aspect ratios for target objects [30]. Anchors are regression references and classification candi-

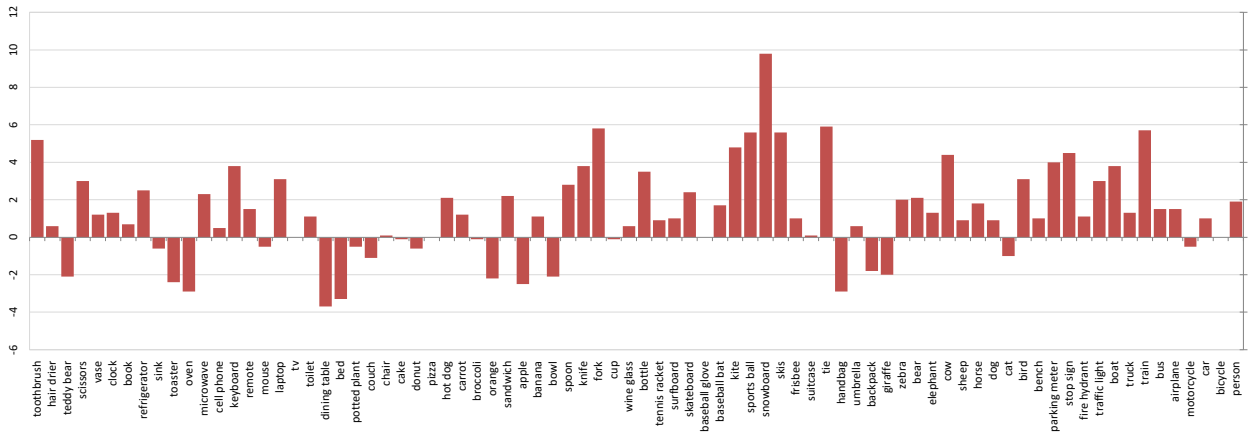


Figure 2. AP difference of FoveaNet and RetinaNet on COCO dataset. Both models use ResNet-FPN-50 as backbone and 800 input scales. FoveaBox shows improvement in most of the classes, especially for classes whose bounding boxes are likely to be more arbitrary, such as *toothbrush, fork, snowboard, tie* and *train*.

dates to predict proposals for two-stage detectors (Faster R-CNN [40], FPN [27]) or final bounding boxes for single-stage detectors (SSD [31], RetinaNet [28]). Nevertheless, the anchor boxes can be regarded as a feature-sharing sliding window scheme to cover the possible locations of objects.

However, the utilization of anchor boxes or (candidate boxes) has some drawbacks. First, anchor boxes introduce additional hyper-parameters of design choices. One of the most important factors in designing anchor boxes is how densely it covers the location space of the targets. To achieve a good recall rate, the anchors are carefully designed based on the statistics computed from the training/validation set. Second, one design choice based on a particular dataset is not always applicable to other applications, which harms the generality[46]. For example, anchors are usually square for face detection. While pedestrian detection needs more tall anchors. Third, since there is a large set of candidate object locations regularly sampled across an image, dense object detectors usually rely on effective techniques to deal with the foreground-background class imbalance challenge [28][23][41].

One choice to improve the anchor generation process is to make it more flexible. Most recently, there are some successful works trying to improve the capacity of the anchor boxes [46][45][49]. In MetaAnchor [46], anchor functions are dynamically generated from the arbitrary customized prior boxes. The Guided-Anchoring method [45] jointly predicts the locations where the center of objects are likely to exist as well as the scales and aspect ratios at different positions. In [49], the authors also suggest dynamically learn the anchor shapes. Nevertheless, these works still rely on the enumeration of possible scales and aspect ratios for the optimization of the model. In MetaAnchor, the input of the anchor functions is the regular sampled anchors with different aspect ratios and scales. In Guided-Anchoring, the authors assume that the center of each anchor is fixed and sample multiple pairs of (w, h) to approximate the best shape centered at the corresponding position.

In contrast, human vision system can recognize the locations of instance in space and predict the boundary given the visual cortex map, without any pre-defined shape template [1]. In other words, we human naturally recognize the object in the visual scene without enumerating the candidate boxes. Inspired by this, an intuitive question to ask is, *is the anchor boxes scheme the optimal way to guide the search of the objects*? If the answer is no, *could we design an accurate object detection framework without the dependence of anchors or candidate boxes*? Without candidate anchor boxes, one might expect a complex method is required to achieve comparable results. However, we show that a surprisingly simple and flexible system can match the prior state-of-the-art object detection performance without the requirement of candidate boxes.

To this end, we present FoveaBox, a completely anchor-free framework for object detection. FoveaBox is motivated from the fovea of human eyes: the center of the vision field (object) is with the highest visual acuity. FoveaBox jointly predicts the locations where the object's center area is likely to exist as well as the bounding box at each valid location. Thanks to the feature pyramidal representations [27], different scales of objects are naturally detected from multiple levels of features. To demonstrate the effectiveness of the proposed detection scheme, we combine the recent progress of feature pyramid networks and our detection head to form the framework of FoveaBox. Without bells and whistles, FoveaBox gets state-of-the-art single-model results on the COCO object detection task [29]. Our best single model, based on a ResNeXt-101-FPN backbone, achieves a COCO `test-dev` AP of 42.1, surpassing most of previously pub-

lished anchor based single-model results.

Since FoveaBox does not rely on the default anchors both at training phase and inference, it is more robust to bounding box distributions. To verify this, we manually elongate the images as well as the annotations of the validation set, and compare the robustness of FoveaBox with the previous anchor-based models [28]. Under this setting, FoveaBox outperforms the anchor-based methods by a large margin. We believe the simple training/inference manner of FoveaBox, together with the flexibility and accuracy, will benefit future research on object detection and relevant topics.

2. Related work

Classic Object Detectors: Prior to the success of deep CNNs, the widely used detection systems are based on the combination of independent components (HOG [5], SIFT [33] etc.). DPM [8] and its variants help extending object detectors to more general object categories and have leading results for many years [6]. The sliding-window approach is the leading detection paradigm for searching the object of interest in classic object detection frameworks.

Modern Object Detectors: Modern object detectors are generally grouped into two factions: two-stage, proposal driven detectors and one-stage, proposal free methods. For two-stage detectors, the first stage generates a sparse set of object proposals, and the second stage classifies the proposals as well as refines the coordinates with the sliding window manner. Such pipeline is first demonstrated its effectiveness by R-CNN [12] and is widely used in later two-stage methods [15][11]. In Faster R-CNN [40], the first stage (RPN) simultaneously predicts object bounds and objectness scores at each pre-defined sliding window anchor with a light-weight network. Several attempts have been performed to boost the performance of the detector, including feature pyramid [27][24][21], multiscale [44][34], and object relations [17], etc.

Compared to two-stage approaches, the one-stage pipeline skips object proposal generation and predicts bounding boxes and class scores in one evaluation. Most top one-stage detectors rely on the anchor boxes to enumerate the possible locations of target objects (e.g. SSD [31], DSSD [9], YOLOv2/v3 [38][39], and RetinaNet [28]). In CornerNet [26], the authors propose to detect an object bounding box as a pair of keypoints. CornerNet adopts the Associative Embedding [35] technique to separate different instances. Some prior works share similarities with our work, and we will discuss them in more detail in §5.

3. FoveaBox

FoveaBox is a single, unified network composed of a backbone network and two task-specific subnetworks. The backbone is responsible for computing a convolutional feature map over an entire input image and is an off-the-shelf convolutional network. The first subnet performs per pixel classification on the backbone's output; the second subnet performs bounding box prediction for the corresponding position. While there are many possible choices for the details of these components, we take the RetinaNet's design [28] for simplicity and fair comparison.

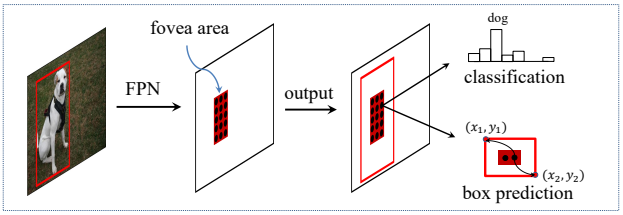


Figure 3. **FoveaBox object detector**. For each output spacial position that potentially presents an object, FoveaBox directly predicts the confidences for all target categories and the bounding box.

3.1. Feature Pyramid Network Backbone

We adopt the Feature Pyramid Network (FPN) [27] as the backbone network for the subsequent detection. In general, FPN uses a top-down architecture with lateral connections to build an in-network feature pyramid from a single-scale input. Each level of the pyramid can be used for detecting objects at a different scale. We construct a pyramid with levels $\{P_l\}, l = 3, 4, \cdots, 7$, where l indicates pyramid level. P_l has $1/2^l$ resolution of the input. All pyramid levels have $C = 256$ channels. For further details about FPN, we refer readers to [27][28].

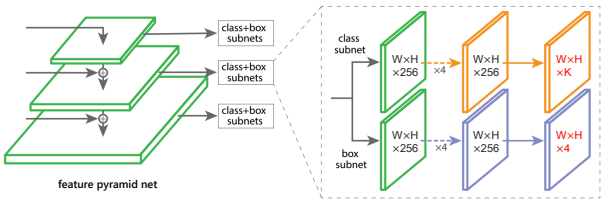


Figure 4. **FoveaBox network architecture**. The design of the architecture follows RetinaNet [28] to make a fair comparison. FoveaBox uses a Feature Pyramid Network [27] backbone on top of a feedforward ResNet architecture. To this backbone, FoveaBox attaches two subnetworks, one for classifying the corresponding cells and one for predict the (x_1, y_1, x_2, y_2) of ground-truth object boxes. For each spatial output location, the FoveaBox predicts one score output for each class and the corresponding 4-dimensional box, which is different from previous works attaching A anchors in each position (usually $A = 9$).

3.2. Scale Assignment

While our goal is to predict the boundary of the target objects, directly predicting these numbers is not stable, due to the large scale variations of the objects. Instead, we divide the scales of objects into several bins, according to the number of feature pyramidal levels. Each pyramid has a basic area ranging from 32^2 to 512^2 on pyramid levels P_3 to P_7, respectively. So for level P_l, the basic area S_l is computed by

$$S_l = 4^l \cdot S_0. \tag{1}$$

Analogous to the ResNet-based Faster R-CNN system that uses C4 as the single-scale feature map, we set S_0 to 16 [40]. Within FoveaBox, each feature pyramid learns to be responsive to objects of particular scales. The valid scale range of the target boxes for pyramid level l is computed as

$$[S_l/\eta^2, S_l \cdot \eta^2], \tag{2}$$

where η is set empirically to control the scale range for each pyramid. Target objects not in the corresponding scale range are ignored during training. Note that an object may be detected by multiple pyramids of the networks, which is different from previous practice that maps objects to only one feature pyramid [27][14].

3.3. Object Fovea

Each output set of pyramidal heatmaps has K channels, where K is the number of categories, and is of size $H \times W$ (Fig. 4). Each channel is a binary mask indicating the possibility for a class. Given a valid ground-truth box denoted as (x_1, y_1, x_2, y_2). We first map the box into the target feature pyramid P_l with stride 2^l

$$\begin{aligned} x_1^{'} &= \frac{x_1}{2^l}, \;\; y_1^{'} = \frac{y_1}{2^l}, \\ x_2^{'} &= \frac{x_2}{2^l}, \;\; y_2^{'} = \frac{y_2}{2^l}, \\ c_x^{'} &= x_1^{'} + 0.5(x_2^{'} - x_1^{'}), \\ c_y^{'} &= y_1^{'} + 0.5(y_2^{'} - y_1^{'}). \end{aligned} \tag{3}$$

The positive area (fovea) $R^{pos} = (x_1^{''}, y_1^{''}, x_2^{''}, y_2^{''})$ of the quadrangle on the score map is designed to be roughly a shrunk version of the original one (see Fig. 3):

$$\begin{aligned} x_1^{''} &= c_x^{'} - 0.5(x_2^{'} - x_1^{'})\sigma_1, \\ y_1^{''} &= c_y^{'} - 0.5(y_2^{'} - y_1^{'})\sigma_1, \\ x_2^{''} &= c_x^{'} + 0.5(x_2^{'} - x_1^{'})\sigma_1, \\ y_2^{''} &= c_y^{'} + 0.5(y_2^{'} - y_1^{'})\sigma_1, \end{aligned} \tag{4}$$

where σ_1 is the shrunk factor. Each cell inside the positive area is annotated with the corresponding target class label for training. For the definition of negative samples, we introduce another shrunk factor σ_2 to generate R^{neg} using Eq.(4). The negative area is the whole feature map excluding area in R^{neg}. If a cell is unassigned, it will be ignored during training. The positive area usually accounts for a small portion of the whole feature map, so we use Focal Loss [28] to train the target L_{cls} of this branch.

3.4. Box Prediction

The object fovea only encodes the existence possibility of the target objects. To decide the location, the model must predict the bounding box for each potential instance. Each ground-truth bounding box is specified in the way $G = (x_1, y_1, x_2, y_2)$. Our goal is to learn a transformation that maps the networks localization outputs $(t_{x_1}, t_{y_1}, t_{x_2}, t_{y_2})$ at cell (x, y) in the feature maps to the ground-truth box G:

$$\begin{aligned} t_{x_1} &= \log \frac{2^l(x + 0.5) - x_1}{z}, \\ t_{y_1} &= \log \frac{2^l(y + 0.5) - y_1}{z}, \\ t_{x_2} &= \log \frac{x_2 - 2^l(x + 0.5)}{z}, \\ t_{y_2} &= \log \frac{y_2 - 2^l(y + 0.5)}{z}, \end{aligned} \tag{5}$$

where $z = \sqrt{S_l}$ is the normalization factor to project the output space to space centered around 1, leading to an easier and stable learning of the target. This function first maps the coordinate (x, y) to the input image, then computes the normalized offset between the projected coordinate and G. Finally the targets are regularized with the log-space function.

For simplicity, we adopt the widely used Smooth L_1 loss [40] to train the box prediction L_{box}. After the targets being optimized, we can generate the box boundary for each positive cell (x, y) on the output feature maps. We note that Eq.(5) and the inverse transformation can be easily implemented by an element-wise layer in modern deep learning frameworks [36][4].

3.5. Optimization

FoveaBox is trained with stochastic gradient descent (SGD). We use synchronized SGD over 4 GPUs with a total of 8 images per minibatch (2 images per GPU). Unless otherwise specified, all models are trained for 270k iterations with an initial learning rate of 0.005, which is then divided by 10 at 180k and again at 240k iterations. Weight decay of 0.0001 and momentum of 0.9 are used. In addition to the standard horizontal image flipping, we also utilize random aspect ratio jittering to reduce the over-fitting. We set $\sigma_1 = 0.3, \sigma_2 = 0.4$ when defining R^{pos} and R^{neg}. Each cell inside R^{neg} is annotated with the corresponding location target for bounding box training.

method	#sc	#ar	AP	AP_{50}	AP_{75}
RetinaNet	1	1	30.2	49.0	31.7
RetinaNet	2	1	31.9	50.0	34.1
RetinaNet	3	1	31.9	49.4	33.8
RetinaNet	1	3	32.4	52.4	33.9
RetinaNet	2	3	34.2	53.1	36.5
RetinaNet	3	3	34.2	53.2	36.9
RetinaNet	4	3	33.9	52.1	36.2
FoveaBox	n/a	n/a	**36.0**	**55.2**	**37.9**

(a) **Varying anchor density of RetinaNet [28] and FoveaBox**

method	AP	$AP_{u<3}$	$AP_{3\leq u<5}$	$AP_{u\geq 5}$
RetinaNet	34.2	36.5	24.5	10.2
RetinaNet*	35.3	**37.1**	25.3	16.3
FoveaBox	**36.0**	**37.1**	**27.4**	**18.4**

(b) **Detection performance with different aspect ratios**

method	backbone	AR_{100}	AR_{300}	AR_{1000}
RPN	ResNet-50	44.5	51.1	56.6
FoveaBox	ResNet-50	**53.0**	**57.5**	**61.8**

(c) **Region proposal results**

Table 1. **Ablation experiments for FoveaBox**. All models are trained on MS COCO `trainval35k` and tested on `minival`. If not specified, ResNet-50-FPN backbone and a 600 pixel train and test image scale are used to do the ablation study. (a) FoveaBox and varying anchor density of RetinaNet. In RetinaNet, increasing beyond 9 anchors does not shown further gains, while FoveaBox could get much better result without anchor enumeration. (c) With the same exact network, FoveaBox is more robust at higher object aspect ratio ($\min(\frac{h}{w}, \frac{w}{h})$) thresholds. (c) FoveaBox could also generate high quality region proposals when changing the optimization target to a class-agnostic head.

3.6. Inference

During inference, we first use a confidence threshold of 0.05 to filter out predictions with low confidence. Then, we select the top 1000 scoring boxes from each prediction layer. Next, non-maximum suppression (NMS) with threshold 0.5 are applied for each class separately. Finally, the top-100 scoring predictions are selected for each image. This inference setting is exactly the same as that in Detectron baseline [13]. Although there are more intelligent ways to perform post-processing, such as bbox voting [10], Soft-NMS [2] or test-time image augmentations, in order to keep simplicity and to fairly compare against the baseline models, we do not use those tricks here.

4. Experiments

We present experimental results on the bounding box detection track of the challenging COCO benchmark [29]. For training, we follow common practice [14][28] and use the COCO `trainval35k` split (union of 80k images from `train` and a random 35k subset of images from the 40k image `val` split). We report lesion and sensitivity studies by evaluating on the `minival` split (the remaining 5k images from val). For our main results, we report COCO AP on the `test-dev` split, which has no public labels and requires use of the evaluation server.

4.1. Ablation Study

Various anchor densities and FoveaBox: One of the most important design factors in a anchor-based detection system is how densely it covers the space of possible image boxes. As anchor-based detectors use a fixed sampling grid, a popular approach for achieving high coverage of boxes in these approaches is to use multiple anchors [27][28] at each spatial position to cover boxes of various scales and aspect ratios. One may expect that we can always get better performance when attaching denser anchors on each position. To verify this assumption, we sweep over the number of scale and aspect ratio anchors used at each spatial position and each pyramid level in RetinaNet, including a single square anchor at each location to 12 anchors per location (Table.1(a)). Increasing beyond 6-9 anchors does not show further gains. The saturation of performance *w.r.t.* density implies the handcrafted, over-density anchors do not offer an advantage.

Over-density anchors not only increase the foreground-background optimization difficulty, but also likely to cause the *ambiguous* position definition problem. For each output spatial location, there are A anchors whose labels are defined by the IoU with the ground-truth. Among them, some of the anchors are defined as positive samples, while others are negatives. However they are sharing the same input features. The classifier needs to not only distinguish the samples from different positions, but also different anchors at the same position.

In contrast, *FoveaBox explicitly predicts one target at each position and gets no worse performance than the best anchor-based model*. The label of the target is defined by if it is inside an object's bounding box. Compare with the anchor based scheme, FoveaBox enjoys several advantages. (a) Since we only predict one target at each position, the output space has been reduced to $1/A$ of the anchor-based method, where A is the anchor number at each position. Since the foreground-background classification challenge has been mitigated, it is easier for the solver to optimize the model. (b) There is no ambiguous problem and the optimization target is more straightforward. (c) FoveaBox is more flexible, since we do not need to extensively design the anchors to see a relatively better choice.

Analysis of Scale Assignment: In Eq.(2), η controls the scale assignment extent for each pyramid. When $\eta = \sqrt{2}$,

the object scales are divided into non-overlapping bins, and each bin is predicted by the corresponding feature pyramid. As η increases, each pyramid will response to more scales of objects. Table 2 shows the impact of η on the final detection performance. We set $\eta = 2$ for all other experiments.

η	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
1.5	35.0	54.4	36.7	17.8	38.9	48.3
2.0	**36.0**	**55.2**	**37.9**	**18.6**	39.4	**50.5**
2.5	35.7	**55.2**	37.6	18.3	**39.7**	49.3
3.0	35.5	54.5	37.5	17.3	40.4	49.9
4.0	34.6	53.0	36.5	16.2	39.8	48.9

Table 2. Varying η for FoveaBox.

FoveaBox is more robust to box distributions: Unlike the traditional predefined anchor strategy, one of the major benefits of FoveaBox is the robust prediction of bounding boxes. To verify this, we conduct two experiments to compare the localization performance of different methods.

In the first experiment, we divide the boxes in the validation set into three groups according to the ground-truth aspect ratios $U = \{u_i = \min(\frac{h_i}{w_i}, \frac{w_i}{h_i})\}, i = 1, \cdots, N$, where N is the instance number in the dataset. We compare FoveaBox and RetinaNet at different aspect ratio thresholds, as shown in Table 1(b). Here $*$ means training the model with aspect ratio jittering. We see that both methods get best performance when u is low. Although FoveaBox also suffers performance decrease when u increases, it is much better than the anchor-based RetinaNet.

To further validate the robustness to bounding boxes of different methods, we manually stretch the image as well as the annotations in the validation set, and exam the behaviors of different detectors. Fig. 5 shows the localization performance at different h/w stretching thresholds. Under the evaluation criterion that $h/w = 1$, the performance gap between the two detectors is relatively small. The gap starts to increase as we increase the stretching threshold. Specifically, FoveaBox gets 21.3 AP when stretching h/w by 3 times, which is 3.7 points better than RetinaNet* counterpart.

The anchor-based methods rely on box regression with anchor reference to generate the final bounding box. In practice, the regressor is trained for the positive anchors, which will harm the generality when predicting more arbitrary shapes of the targets. In FoveaBox, each prediction position is not associated with the particular reference shape, and it directly predicts the target ground-truth boxes. Since FoveaBox allows arbitrary aspect ratios, it is capable of capturing those extremely tall or wide objects better. See some qualitative examples in Fig. 6.

Per-class difference: Fig. 2 shows per-class AP difference of FoveaBox and RetinaNet. Both of them are with ResNet-50-FPN backbone and 800 input scale. The vertical axis shows $AP_{FoveaBox}$-$AP_{RetinaNet}$. FoveaBox shows improvement in most of the classes, especially for classes whose bounding boxes are likely to be more arbitrary. For class *toothbrush, fork, sports ball, snowboard, tie* and *train*, the AP improvements are larger than 5 points.

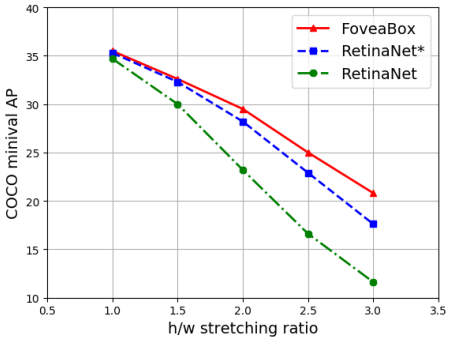
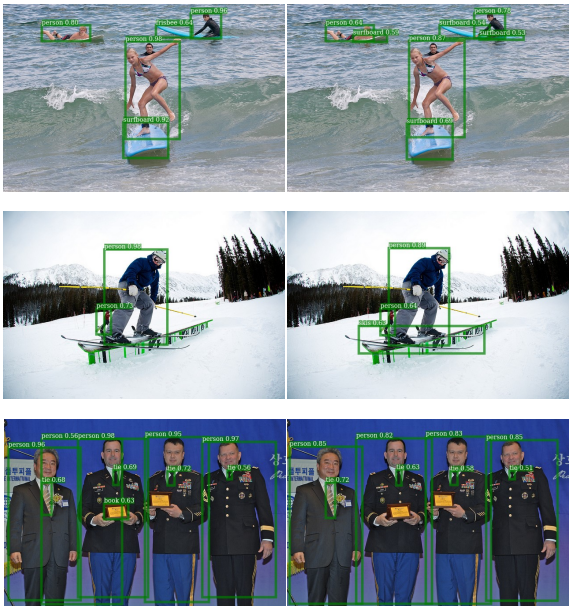


Figure 5. **Evaluating boxes at different h/w stretching thresholds**. * means training the model with aspect ratio jittering. Since we see similar trends when stretching w/h, we only show h/w stretching results here.



(a) (b)

Figure 6. **Qualitative comparison** of RetinaNet (a) and FoveaBox (b). Our model shows improvements in classes with large aspect ratios. Better viewed in color.

Generating high-quality region proposals: Changing the classification target to class-agnostic head is straightforward and could generate region proposals. We compare the proposal performance against region proposal network (RPN) [40] and evaluate average recalls (AR) with different numbers of proposals on COCO `minival` set, as shown in

Figure 7. **FoveaBox results on the COCO `minival` set**. These results are based on ResNet-101, achieving a single model box AP of 38.9. For each pair, left is the detection results with bounding box, category, and confidence. Right is the score output map with their corresponding bounding boxes before feeding into non-maximum suppression (NMS). The score probability in each position is denoted by the color density. These figures demonstrate that FoveaBox could directly generate accurate, robust box predictions, without the requirement of candidate anchors.

Table 1(c).

Surprisingly, our method outperforms the RPN baseline by a large margin, among all criteria. Specifically, with top 100 region proposals, FoveaBox gets 53.0 AR, outperforming RPN by 8.5 points. This validates that our model's capacity in generating high quality region proposals.

Across model depth and scale: Table 3 shows FoveaBox utilizing different backbone networks and input resolutions. The inference settings are exectly the same as RetinaNet, and the speed is also on par with the corresponding baseline.

	depth	scale	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
RetinaNet	50	400	30.5	47.8	32.7	11.2	33.8	46.1
FoveaBox	50	400	31.9	49.3	33.8	12.7	36.1	48.7
RetinaNet	50	600	34.2	53.2	36.9	16.2	37.4	47.4
FoveaBox	50	600	36.0	55.2	37.9	18.6	39.4	50.5
RetinaNet	50	800	35.7	55.0	38.5	18.9	38.9	46.3
FoveaBox	50	800	37.1	57.2	39.5	21.6	41.4	49.1
RetinaNet	101	400	31.9	49.5	34.1	11.6	35.8	48.5
FoveaBox	101	400	33.3	51.0	35.0	12.9	38.0	51.3
RetinaNet	101	600	36.0	55.2	38.7	17.4	39.6	49.7
FoveaBox	101	600	38.0	57.8	40.2	19.5	42.2	52.7
RetinaNet	101	800	37.8	57.5	40.8	20.2	41.1	49.2
FoveaBox	101	800	38.9	58.4	41.5	22.3	43.5	51.7

Table 3. FoveaBox across different input resolutions and model depths.

As shown in Table 3, FoveaBox improves on RetinaNet baselines consistently by $1 \sim 2$ points. When analysing the performances on small, medium and large object scales, we observe that the improvements come from all scales of objects.

4.2. Main Results

We compare FoveaBox to the state-of-the-art methods in object detection in Table 4. All instantiations of our model outperform baseline variants of previous state-of-the-art models. The first group of detectors on Table 4 are two-stage detectors, the second group one-stage detectors, and the last group the FoveaBox detector. FoveaBox outperforms all single-stage detectors under ResNet-101 backbone, under all evaluation metrics. This includes the very recent one-stage CornerNet [26]. FoveaBox also outperforms most two-stage detectors, including FPN [27] and Mask R-CNN [14].

Two-stage detectors rely on region-wise sub-networks to further classify the sparse region proposals. Cascade R-CNN extends the two-stage scheme to multiple stages to further refine the regions. Since FoveaBox could also generate region proposals by changing the model head to class agnostic scheme (Table 1(c)), we believe it could further improve the performance of two-stage detectors, which beyond the focus of this paper.

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
two-stage							
Faster R-CNN+++[16]*	ResNet-101	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN by G-RMI[19]	Inception-ResNet-v2	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w FPN[27]	ResNet-101	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN w TDM[42]	Inception-ResNet-v2	36.8	57.7	39.2	16.2	39.8	52.1
Mask R-CNN[14]	ResNet-101	38.2	60.3	41.7	20.1	41.1	50.2
Relation Network [17]	DCN-101	39.0	58.6	42.9	-	-	-
IoU-Net [20]	ResNet-101	40.6	59.0	-	-	-	-
Cascade R-CNN [3]	ResNet-101	42.8	62.1	46.3	23.7	45.5	55.2
one-stage							
YOLOv2 [37]	DarkNet-19	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [9]	ResNet-101	31.2	50.4	33.3	10.2	34.5	49.8
YOLOv3 [39]	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9
DSSD513 [9]	ResNet-101	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet [28]	ResNet-101	39.1	59.1	42.3	21.8	42.7	50.2
ConRetinaNet [22]	ResNet-101	40.1	59.6	43.5	23.4	44.2	53.3
CornerNet[26]*	Hourglass-104	40.5	56.5	43.1	19.4	42.7	53.9
RetinaNet [28]	ResNeXt-101	40.8	61.1	44.1	24.1	44.2	51.2
ours							
FoveaBox	ResNet-101	40.6	60.1	43.5	23.3	45.2	54.5
FoveaBox	ResNeXt-101	42.1	61.9	45.2	24.9	46.8	55.6

Table 4. **Object detection** single-model results *v.s.* state-of-the-arts on COCO `test-dev`. We show results for our FoveaBox models with 800 input scale. Both RetinaNet and FoveaBox are trained with scale jitter and for 1.5× longer than the same model from Table 3. Our model achieves top results, outperforming all one-stage and most two-stage models. The entries denoted by "*" use bells and whistles at inference.

5. More Discussions to Prior Works

Before closing, we investigate some relations and differences between FoveaBox and some prior works.

Score Mask for Text Detection: The score mask technique has been widely used in the area of text detection [48][18][50]. Such works usually utilize the fully convolutional networks [32] to predict the existence of target scene text and the quadrilateral shapes. Compared with scene text detection, generic object detection is more challenging since it faces more occlusion, multi-class classification and scale problems. Naively adopting the text detection methods into generic object detection usually gets poor performances.

Guided-Anchoring [45]: It jointly predicts the locations where the center of objects of interest are likely to exist as well as the scales and aspect ratios centered at the corresponding locations. If (x, y) is not in the target center, the detected box will not be the optimal one. Guided-Anchoring relies the center points to give the best predictions. In contrast, FoveaBox predicts the (*left, top, right, bottom*) boundaries of the object for each foreground position, which is more robust.

FSAF [51]: This is a *contemporary* work with FoveaBox. It also tries to directly predict the bounding boxes of the target objects. The differences between FoveaBox and FSAF are: (a) FSAF relies online feature selection module to select suitable features for each instance and anchors. While in FoveaBox, instance of a particular scale is simultaneously optimized by adjacent pyramids, determined by Eq.(2), which is more simple and robust. (b) For the optimization of box boundary, FSAF utilizes the IoU-Loss [47] to maximize the IoU between the predicted box and groundtruth. While in FoveaBox, we use the Smooth L_1 loss to directly predict the four boundaries, which is more simple and straightforward. (c) FoveaBox shows much better performance compared with FSAF, as shown in Table 5.

method	backbone	AP	AP_{50}	AP_{75}
FSAF	ResNet-50	35.9	55.0	37.9
FSAF+Retina	ResNet-50	37.2	57.2	39.4
FoveaBox	ResNet-50	37.1	57.2	39.5
FoveaBox+Retina	ResNet-50	38.1	57.8	40.5

Table 5. Performance comparison of the contemporary work FSAF [51] and FoveaBox. '+Retina' means combining the results of the relevant methods and the outputs of anchor-based RetinaNet.

CornerNet [26]: CornerNet proposes detecting objects by the top-left and bottom-right keypoint pairs. The key step of CornerNet is to recognize which keypoints belong to the same instance and grouping them correctly. In contrast, the instance class and the bounding box are associated together in FoveaBox. We directly predict the boxes and classes, without any grouping scheme to separate different instances.

6. Conclusion

We have presented FoveaBox for generic object detection. By simultaneously predict the object position and the

corresponding boundary, FoveaBox gives a clean solution for detecting objects without prior candidate boxes. We demonstrate its effectiveness on standard benchmarks and report extensive experimental analysis.

References

[1] M. F. Bear, B. W. Connors, and M. A. Paradiso. *Neuroscience*, volume 2. Lippincott Williams & Wilkins, 2007.

[2] N. Bodla, B. Singh, R. Chellappa, and L. 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] Z. Cai and N. Vasconcelos. Cascade r-cnn: Delving into high quality object detection. *arXiv preprint arXiv:1712.00726*, 2017.

[4] H.-Y. Chen et al. Tensorflow–a system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016.

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[6] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[7] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester. Cascade object detection with deformable part models. In *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*, pages 2241–2248. IEEE, 2010.

[8] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2010.

[9] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg. Dssd: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*, 2017.

[10] S. Gidaris and N. 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] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[12] R. Girshick, J. Donahue, T. Darrell, and J. 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] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, and K. He. Detectron. https://github.com/facebookresearch/detectron, 2018.

[14] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE, 2017.

[15] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European conference on computer vision*, pages 346–361. Springer, 2014.

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[17] H. Hu, J. Gu, Z. Zhang, J. Dai, and Y. Wei. Relation networks for object detection. In *Computer Vision and Pattern Recognition (CVPR)*, volume 2, 2018.

[18] H. Hu, C. Zhang, Y. Luo, Y. Wang, J. Han, and E. Ding. Wordsup: Exploiting word annotations for character based text detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4940–4949, 2017.

[19] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *IEEE CVPR*, volume 4, 2017.

[20] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang. Acquisition of localization confidence for accurate object detection. In *Proceedings of the European Conference on Computer Vision, Munich, Germany*, pages 8–14, 2018.

[21] T. Kong, F. Sun, W. Huang, and H. Liu. Deep feature pyramid reconfiguration for object detection. *european conference on computer vision*, pages 172–188, 2018.

[22] T. Kong, F. Sun, H. Liu, Y. Jiang, and J. Shi. Consistent optimization for single-shot object detection. *arXiv preprint arXiv:1901.06563*, 2019.

[23] T. Kong, F. Sun, A. Yao, H. Liu, M. Lu, and Y. Chen. Ron: Reverse connection with objectness prior networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 2, 2017.

[24] T. Kong, A. Yao, Y. Chen, and F. Sun. Hypernet: Towards accurate region proposal generation and joint object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 845–853, 2016.

[25] C. H. Lampert, M. B. Blaschko, and T. Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *2008 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2008.

[26] H. Law and J. Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 6, 2018.

[27] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature pyramid networks for object detection. In *CVPR*, volume 1, page 4, 2017.

[28] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[29] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[30] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen. Deep learning for generic object detection: A survey. *arXiv preprint arXiv:1809.02165*, 2018.

[31] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector.

In *European conference on computer vision*, pages 21–37. Springer, 2016.

[32] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[33] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[34] M. Najibi, B. Singh, and L. S. Davis. Autofocus: Efficient multi-scale inference. *arXiv preprint arXiv:1812.01600*, 2018.

[35] A. Newell, Z. Huang, and J. Deng. Associative embedding: End-to-end learning for joint detection and grouping. In *Advances in Neural Information Processing Systems*, pages 2277–2287, 2017.

[36] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017.

[37] J. Redmon, S. Divvala, R. Girshick, and A. 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.

[38] J. Redmon and A. Farhadi. Yolo9000: better, faster, stronger. *arXiv preprint*, 2017.

[39] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[40] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[41] A. Shrivastava, A. Gupta, and R. Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 761–769, 2016.

[42] A. Shrivastava, R. Sukthankar, J. Malik, and A. Gupta. Beyond skip connections: Top-down modulation for object detection. *arXiv preprint arXiv:1612.06851*, 2016.

[43] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[44] B. Singh and L. 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.

[45] J. Wang, K. Chen, S. Yang, C. C. Loy, and D. Lin. Region proposal by guided anchoring. *arXiv preprint arXiv:1901.03278*, 2019.

[46] T. Yang, X. Zhang, Z. Li, W. Zhang, and J. Sun. Metaanchor: Learning to detect objects with customized anchors. In *Advances in Neural Information Processing Systems*, pages 318–328, 2018.

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

[48] Z. Zhang, C. Zhang, W. Shen, C. Yao, W. Liu, and X. Bai. Multi-oriented text detection with fully convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4159–4167, 2016.

[49] Y. Zhong, J. Wang, J. Peng, and L. Zhang. Anchor box optimization for object detection. *arXiv: Computer Vision and Pattern Recognition*, 2018.

[50] X. Zhou, C. Yao, H. Wen, Y. Wang, S. Zhou, W. He, and J. Liang. East: an efficient and accurate scene text detector. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 5551–5560, 2017.

[51] C. Zhu, Y. He, and M. Savvides. Feature selective anchor-free module for single-shot object detection. *arXiv preprint arXiv:1903.00621*, 2019.