DetOH: An Anchor-Free Object Detector with Only Heatmaps

Ruohao Wu[1], Xi Xiao[1](✉), Guangwu Hu[3], Hanqing Zhao[2], Han Zhang[2], and Yongqing Peng[2]

[1] Shenzhen International Graduate School, Tsinghua University, Shenzhen 518055, China
xiaox@sz.tsinghua.edu.cn
[2] Beijing Research Institute of Telemetry, Beijing 100076, China
[3] Shenzhen Institute of Information Technology, Shenzhen 518172, China

Abstract. In object detection, the anchor-based method relies on too much manual design, and the training and prediction process is too inefficient. In recent years, one-stage anchor-free methods such as Fully Convolutional One-stage Object Detector (FCOS) and CenterNet have made a splash in object detection. They not only have a simple structural design, but also demonstrate competitive performance. They have exceeded many two-stage or anchor-based approaches. However, in industrial applications, the design of its multiple output heads hinders the installation of the model. At the same time, different output heads mean a combination of multiple loss functions. This introduces problems in training. Here, we propose an anchor-free object Detector with Only Heatmaps (DetOH) to solve object detection. Bounding box parameters are calculated by post-processing. The design of the single output head allows object detection to use a semantic segmentation network, realizing the unification of the two frameworks. In addition, compared with CenterNet, we have greatly improved the speed of object detection (6 vs. 32 Frames Per Second) with 3.2% Average Precision boost. The proposed DetOH framework can be applied to multi-target tracking, key point detection and other tasks.

Keywords: Heatmaps · Anchor-free · Object Detector

1 Introduction

Object detection is an algorithm that predicts the bounding box position and category label for each instance of interest in an image. The classical algorithms mainly rely on sliding windows [1, 2], which classify every possible position and therefore require high speed. This also established the position of the anchor in object detection. After the advent of deep learning, detection has shifted to the use of FCN (Fully Convolutional Networks) since Faster R-CNN [3]. Many current anchor-based detectors such as Faster R-CNN, SSD [4], YOLOv2 [5], and v3 [6] rely on a predefined set of anchor boxes.

Many anchor-free detectors have also appeared in recent years, and their performance has gradually surpassed that of anchor-based detectors. For example, CenterNet [7, 8] predicts promising performance with hourglass for predicting center, offsets, and object size. Fully Convolutional One-stage Object Detector (FCOS) [9, 10] uses a FCN to demonstrate the commonality between semantic segmentation tasks and object detection tasks. However, compared with the simple output head and loss function of semantic segmentation, an anchor-free detector requires multiple loss function combinations to assist training due to its multiple outputs. It is natural to ask the question: can we really do object detection like semantic segmentation? The answer is yes.

We found that the reason for the multiple outputs of the anchor-free detector is the deviation of the center positioning and the parameters of the object box. The former is caused by the output feature map being smaller than the input. The latter can be solved by mining information in classification heatmaps. Therefore, we designed the polynomial heatmaps so that the boundary of the object box can be obtained by post-processing. At the same time, their size is equal to the input, thus avoiding the offset of the center positioning. To highlight the portability and speed/precision balance of our model, we applied it to a smart security system. The detailed contributions are as follows.

1) We proposed an anchor-free object detector with only heatmaps. This not only simplifies the process of object detection but also allows for greater flexibility in detecting objects of varying sizes and shapes. We also introduced the concept of polynomial heatmaps for object detection which helps in post-processing and precise predictions of object boundaries.
2) We designed the Center Point (CP) loss function to effectively improve the detector performance. It draws on the positioning of the CP in the inference process and effectively alleviates the problem of CP offset. This greatly improves model accuracy.
3) We conducted experiments on the Microsoft COCO (Common Objects in COntext) dataset and applied the model to embedded devices to demonstrate the effectiveness of our work. Compared to CenterNet that use the same backbone network, we improved model performance by 3.2% Average Presicion (AP), increasing speeds from 6 to 32 FPS (Frames Per Second).

This paper is organized as follows. Related work and methods are analyzed in Sect. 2. In Sect. 3, the specific methods and core innovations are introduced. The experiments are shown in Sect. 4 followed by the ablation study in Sect. 5.

2 Related Work

In this section, we introduce some work related to our method. They are mainly divided into two parts: anchor-based detectors and anchor-free detectors.

2.1 Anchor-Based Detectors

Many anchor-based detectors achieve a good balance of speed and accuracy, such as Fast Region-based Convolutional Network (R-CNN) [13] and Deconvolutional Single Shot Detector (DSSD) [15]. Anchor boxes can be considered suggested regions, and they are classified as correct or negative patches. The anchor utilizes and avoids duplicate feature calculations, greatly speeding up the detection process. But it is worth noting that anchor-based detectors have some drawbacks:

As shown by Faster R-CNN [3] and RetinaNet [11], the performance of the detection is sensitive to the size, proportion, and number of anchor boxes. These hyperparameters can greatly affect the AP performance on the COCO dataset [12]. Therefore, these hyperparameters need to be fine-tuned in anchor-based detectors.

Detectors difficulty handling object candidates with large shape variations, especially small objects. Therefore, they need to set different sizes or aspect ratios for the same kind of object. To achieve high accuracy, anchor boxes need to be densely placed on the input image. Meanwhile, a lot of negative samples are generated, which brings imbalance problems to training. Anchor boxes also introduce complex computations into the training process and loss functions, such as calculating IoU (Intersection over Union).

2.2 Anchor-Free Detectors

The earliest anchor-free detector was probably YOLOv1 [6], which predicts points near the center of the object's bounding box because they are believed to be able to produce higher quality detections. But using only points close to the center resulted in low recall. CornerNet [16] uses a pair of corner points to detect boundaries and groups them to form the final detected box. CornerNet learns an additional distance metric, the purpose of which is to find pairs of corner points belonging to the same instance. This requires more complex post-processing. Another detector, Unitbox [17], is based on DenseBox [18]. Unitbox is considered unsuitable for general object detection because of the difficulty of handling overlapping bounding boxes and the relatively low recall. Feature Selective Anchor-Free (FSAF) [19] proposes to add an anchor-based detection branch to anchor-free detectors. As they consider that completely anchor-free detectors do not achieve good performance, they also utilize feature selection modules to improve the performance of anchor-free branching. So anchor-free detectors have comparable performance to anchor-based detectors. RepPoints [20] indicates that the box consists of a set of points and uses a conversion function to obtain the object box. Corner Proposal Network (CPN) [21] and HoughNet [22] require grouping or post-voting processing, so they are quite complex and slow. CenterNet [7] is a concurrent anchor-free detector. It uses a clean network structure to demonstrate the performance to be expected. A similar model, FCOS [9] adds center-ness branching, enabling a better accuracy/speed trade-off. There are many subsequent work based on FCOS [25, 26]. These models enhance the detection characteristics, loss function or allocation strategy of FCOS to further improve the performance of anchor-free detectors.

However, these detectors often have complex output heads which bring trouble to model training and loss function design. Besides, its multiple output heads are usually not supported by embedded chips, making it difficult for industrial applications.

3 Our Approach

In this section, our approach is divided into four parts. The model overview is first introduced, followed by our heatmaps and the calculation of the object size, and finally the loss function in our work.

3.1 Model Overview

In this paper, we propose an anchor-free object detection method with only heatmaps. Similar to other object detectors that use heatmaps, our model is divided into two parts, the feature extraction backbone network and the output head. The backbone network is mainly used to extract the image features of various scales, mostly using mature convolutional networks, such as ResNet [28], Deep Layer Aggregation (DLA) [29] and Hourglass [30]. FPN (Feature Pyramid Network) [23] or BiFPN [24] are sometimes added. The output head converts the feature map extracted by the backbone into a feature map that can obtain object box information. This is the main difference between these detectors. CenterNet and FCOS are the most representative models that use heatmaps for object detection.

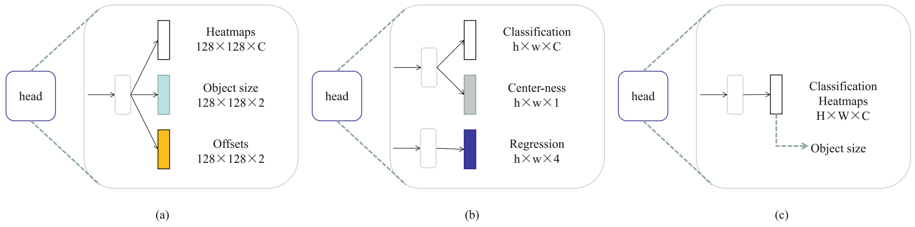

Fig. 1. Output head structure in different models. (a) shows CenterNet's output head, the prediction module that contains Heatmaps, Object size, and Offsets. (b) shows the output head of FCOS that contains Classification, Center-ness and Regression at different scales. (c) shows the output head of our method, the prediction module only contains Classification Heatmaps.

As shown in the Fig. 1, CenterNet's output head, i.e., the prediction module, contains three branches: Heatmaps, Object size, and Offsets. The center of the object is obtained through the heatmaps, and then the offset is corrected moderately. Finally, the size of the object is added to get a complete object box. For an input image of 512 × 512 × 3 pixels, the output heatmaps size is 128 × 128 × C pixels, where C is the number of categories.

The output head of FCOS also contains three parts, i.e., Classification, Center-ness and Regression. The center of the object is obtained through the product feature maps of classification and center-ness, and then the offset and size information are obtained by regression. Then a complete object box is obtained, which utilizes a feature pyramid structure and thus contains several output heads of multiple scales. The size of the feature map varies from 1/8 to 1/128 of the input image.

However, the output head of our model only contains Classification Heatmaps, from which the object size can be calculated by post-processing. It is worth noting that the classification heatmaps and input images are of equal size.

We design the heatmaps to be as large as the input image, so that we can directly get to the object CP without offsets according to the heatmaps. Meanwhile, unlike the Gaussian circles generated by CenterNet, we plan to generate polynomial heatmaps. This means that a heat spot representing an object is no longer isotropic. The rate at which it decays in different directions is related to its dimensions in the corresponding direction, so that the width and height can be calculated from the parameters of the polynomial. Thus, the output of object size feature map is not required. After removing the object size feature map and the offset feature map, the model only needs one branch, i.e., the classification heatmaps. This greatly simplifies the model.

This simplification is meaningful in many aspects. First, it can support more embedded devices. Some embedded chips do not support models with multiple output heads, which causes them to encounter some obstacles in industrial applications. Secondly, its training mode is end-to-end, which can directly calculate the loss function between the output value and the predicted value. Many models design different loss functions for different output heads. And different positive and negative samples need to be generated to meet their training requirements. This brings a lot of inconvenience to fine tuning in industrial applications. Finally, since the input and output are equally large, the model can be realized by using the FCN. There is little need to limit the size of the input image, which does not need to be square. Moreover, the design of FCN enables the model to use different sizes of pictures in training and testing. This enables high resolution image applications.

3.2 Heatmaps

Before introducing our heatmaps, we can review how CenterNet generates heatmaps.

As CenterNet first filled the input image into a square and resize it to 512 × 512. Its output heatmaps was 1/4 the size of the input after downsampling. Therefore, there was a certain offset error between the position of the center from the heatmaps and the real object center. The offset feature map was set to compensate for the offset error. At the same time, the radius of the Gaussian heat circle was determined by the height and width of the object. Thus, the object size feature map was needed to generate the height and width (Fig. 2).

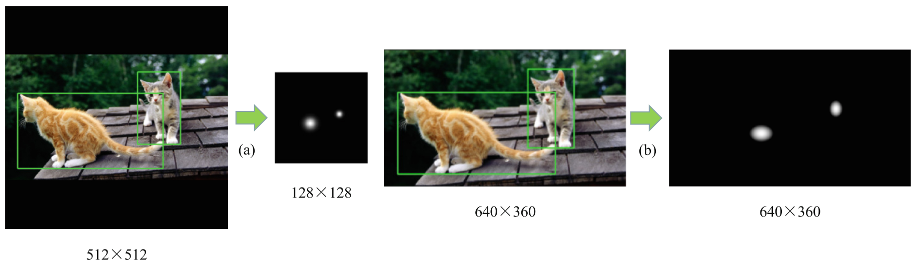

Fig. 2. Heatmap between CenterNet and our method. CenterNet limited the size of the input image to 512 × 512 by padding and resizing. The heatmap size was 128 × 128, and the heat spots of the two objects were circular. Our method does not need to limit the input image size, and the heat map is of the same size. The heat spot shape of an object is related to its horizontal and vertical sizes.

In our method, there is no need to fill the input image into a square. The size of classification heatmaps is set to be equal to the input image, and the position of the center obtained from the heatmaps is the real object center without offset error. So there is no need to output the offset feature map. Meanwhile, the way we generate the heatmaps is different from CenterNet. It used Gaussian distributions in heatmaps, but we use polynomial heatmaps. The radius of the Gaussian circle is jointly determined by the height and width of the object. The polynomial heatmaps in our method can be correlated with the length and width in the corresponding direction. Then the object size feature map is not needed, because the length and width of the object can be calculated according to the polynomial heatmaps.

To calculate the size parameters in both horizontal and vertical directions from the heatmaps, we define a simple polynomial which is related to the width and height of the object. For the object box whose CP is located at (x_c, y_c) and whose width is w and height is h, we define the probability heat value generation mode as polynomial. It can be expressed as,

$$v = 1 - \alpha(\frac{|x - x_c|}{w})^r - \alpha(\frac{|y - y_c|}{h})^r \tag{1}$$

where v represents the probability heat value at the position (x, y), r represents the degree of the polynomial, α represents the size attenuation coefficient, and its value should be greater than 0.5. When its value is 0.5, the probability heat value attenuates to 0 at the midpoint of the edge of the object box. All values greater than 0 form an inscribed graph of the object box. To avoid having a heat value less than zero, we set the value less than 0 to 0, to keep the probability between 0 and 1.

By adjusting r (the degree of the polynomial) and α (the attenuation coefficient) in Eq. (1), we can get different types of heat spots. A few examples of the heatmaps that can be generated with different hyperparameters are shown in Fig. 3.

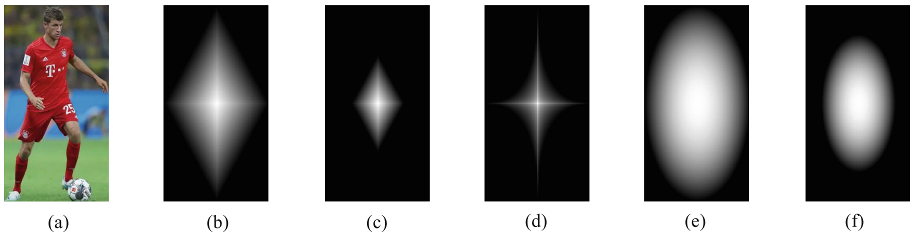

Fig. 3. Examples of the heatmap with different *r* and *α*. The luminance in the figure reflects the magnitude of the probability heat value. (a) is the image slice of an object box; (b) and (c) are the heatmaps when r is 1, but their α are different with values of 0.5 and 1, respectively; (d) is the heatmap when r is 0.5; (e) and (f) are the heatmaps when r is 2, but their α are different with values of 0.5 and 1, respectively.

When two objects are too close, the problem of overlapping heat regions occurs. Our approach is to take the larger value at the overlapping position. This tries to avoid the influence of one on the other. As long as the centers do not coincide exactly, we can locate the centers of both. But when it comes to calculating the size of an object, it might oversize.

3.3 Object Size

The whole process is divided into three parts: Locating the CP of the object; Calculating its width and height according to the heatmaps; Finally, fine-tuning the boundaries of the object.

CP Position. The process for locating the CP of an object is slightly different from CenterNet. Before the maxpooling step, we first perform a large kernel mean filtering on the heatmaps. This is because there are multiple identical maximum values near the CP of the large object, causing duplicate object boxes. After that, it is maxpooled. By comparing the maxpooling result with the mean filtering result, the position with the same value is the local maximum point, which is the center of the object.

Size Parameters. After locating the center of the object box, we need to get the width and height of the object. On the heatmaps, the probability heat value decreases in the corresponding direction from the center of the object. When traversing from the center of the object along the horizontal or vertical direction, the probability heat value change at this time is independent of the coordinates in the other direction. This is easy to get from Eq. (1). According to the mapping relationship between the value and the distance from the CP, the corresponding direction size parameter is determined. The width can be calculated by the following formula:

$$w = \frac{\sqrt[r]{\alpha}}{2}\left(\frac{(x_c - x_l)}{\sqrt[r]{1 - v_l}} + \frac{(x_r - x_c)}{\sqrt[r]{1 - v_r}}\right) \tag{2}$$

where v_l represents the probability heat value at the position (x_l, y_c) on the left of the center, v_r represents the probability heat value at the position (x_r, y_c) on the right of the center, and α is the attenuation coefficient. When the probability heat value decays to a more reliable value, i.e., the probability threshold such as 0.5, the width of the object frame can be calculated by Eq. (2). Similarly, the height can be calculated by the following formula:

$$\mathrm{h} = \frac{\sqrt[r]{\alpha}}{2}\left(\frac{(y_c - y_t)}{\sqrt[r]{1 - v_t}} + \frac{(y_b - y_c)}{\sqrt[r]{1 - v_b}}\right) \tag{3}$$

where v_t represents the probability heat value at the position (x_c, y_t) on the top of the center, v_b represents the probability heat value at the position (x_c, y_b) below the center, and α is the attenuation coefficient.

Bounding Box. With the CP position and width and height, it is easy to calculate the position of the object's bounding box. However, in the experiment, we found that fine-tuning the bounding box can alleviate the problem of CP positioning bias. For example, when the CP is unbiased, the values of the two terms in parentheses in Eq. (2) should be equal. But when offset, the two are unequal. Taking the left border as an example, its value is considered to be related to the width calculated by the left position (x_l, y_c). The formula is expressed as:

$$b_l = x_c - \frac{\sqrt[r]{\alpha}}{2}\frac{(x_c - x_l)}{\sqrt[r]{1 - v_l}} \tag{4}$$

where b_l represents the left boundary of the object.

This way, when the CP is left off, the left border is skewed to the right accordingly, and vice visa. The other three boundaries can be calculated in a similar way. With all the boundary parameters, we get a complete object box.

3.4 Loss Function

The loss function adopted in this paper is a combination of two loss functions. One is the MSE (Mean Squared Error) loss, and the other is the CP Loss designed for the model proposed in this paper. The linear combination of the two functions is the loss function adopted in the training of our model, which can be expressed by the formula:

$$L(y, p) = a \cdot L_{MSE}(y, p) + L_{CP}(y, p) \tag{5}$$

where y is the ground truth, p is the predicted value, and a is a hyperparameter used to coordinate the difference of orders of magnitude between the two loss functions. In this paper, the value is set to 100.

The reason why the MSE loss is selected is that the probability heat value is continuous. While the classical CE (Cross Entropy) loss function is suitable for the calculation of discrete variables, i.e., the case of only 0 or 1. Similarly, the Dice loss [31] widely used in image segmentation is less applicable to continuous variables. To confirm this, we conducted ablation experiments on the loss function to prove its effectiveness, as detailed in Subsect. 5.2.

During the experiment, we found that if the CP position of a object deviated, it would lead to problems in the calculation of width and height, and thus greatly reducing the accuracy. To alleviate this problem, we propose a CP loss function, which draws on the center position in model inference and the Dice loss design.

The specific method is as follows: Firstly, the maxpooling layer with step size of one is used to process the maximum value of the neighborhood of the output heatmaps. Secondly, the mask matrix is defined as the position of the maximum point. In other words, the position of the pooled heatmaps that is equal to the original heatmaps is assigned the value of 1, and the rest is 0. It can be expressed by the following formula:

$$mask(x, y) = \begin{cases} 0, \ mp(hm(x, y)) \neq hm(x, y) \\ 1, \ mp(hm(x, y)) = hm(x, y) \end{cases} \tag{6}$$

where, $mask(x, y)$ represents the mask matrix value at the position (x, y), $hm(x, y)$ represents the output heatmaps value at the position (x, y), and $mp()$ represents the maxpooling operation, whose kernel size is set to the same as in the locating CP.

After obtaining the mask matrix, we can calculate the cross entropy of the predicted value and the true value after the mask. Specifically, multiplying the predicted value and the true value by the mask matrix and dividing the calculation result of the cross-entropy loss function by the modulus of the mask matrix, we can get the CP loss function in Eq. (7).

$$L_{CP}(y, p) = \frac{L_{CE}(y \cdot mask, p \cdot mask)}{mask} \tag{7}$$

In other words, the CP loss function only considers the error at the maximum point of the classification heatmaps. When the maximum point is closer to the object center, the loss function value is smaller, and vice versa. Meanwhile, it also limits the number of maximum points. Too many maximum values will lead to the increase of 1s of the mask matrix, thus increasing the value of the loss function. The loss function of CP greatly improves the CP positioning, which is also confirmed by the ablation experiments.

4 Evaluation

This section shows the experimental result: firstly, data and experimental environment we adopt and its settings, and then the accuracy and speed of each model.

4.1 Data and Settings

The experimental data set uses the COCO dataset [12]. We use the COCO train2017 split for training and val2017 split as validation for our evaluation study. We report our main results on the test-dev split by uploading our detection results to the evaluation server. All models are trained on the PyTorch 1.9.1 framework and an NVIDIA 2080Ti GPU with 11 GB memory.

4.2 Accuracy and Speed

The accuracy index we use is mAP (mean Average Precision). The speed uses the common metric FPS. AP_{50} refers to the AP when the value of the IoU is 50%, and the same is true for AP_{75}. AP_S, AP_M and AP_L are the AP values of three different scale objects of small, medium and large. The speed/accuracy comparison shows between DetOH and some of the most recent detection methods in Fig. 4.

Using the backbone DLA-34 [29], DetOH can achieve 40.3% of AP at 68 FPS on a single 2080Ti GPU graphics card. We further replaced DLA-34 with a deeper network DLA-60, resulting in a better speed/accuracy trade-off (43.3% AP at 32 FPS). Compared to CenterNet [32], we improved network performance by 3.2% AP, increasing speeds from 6 to 32 FPS. This means that with improved accuracy, DetOH is 433% faster than CenterNet when using the same backbone network. DetOH also outperforms other methods in terms of speed and accuracy, including anchor-based methods.

To achieve higher accuracy, we use deeper backbone networks and a more efficient feature extraction structure. The specific results are shown in Table 1. As it can be seen,, our model AP is higher than all classic models. Specifically, it is 1.2% AP higher than CenterNet and 2.9% AP higher than FCOS. It is worth noting that our approach is significantly ahead in the performance of small objects. This may be due to the beneficial gain brought about by increasing the output resolution. Some of the latest methods [27] with a particularly high AP use a lot of tricks, such as data augmentation during the testing phase, increasing deformable convolution.

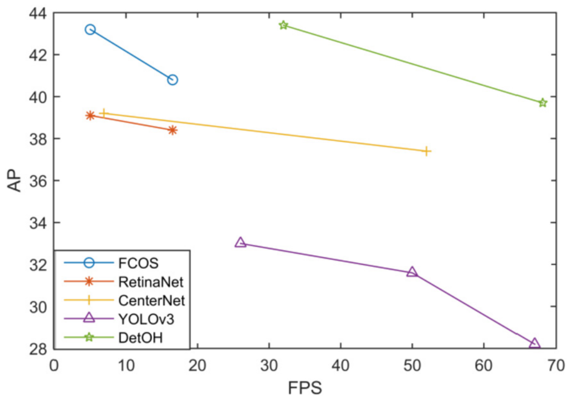

Fig. 4. Speed/accuracy trade-off between DetOH and several recent methods: FCOS [9], CenterNet [7], YOLOv3 [6] and RetinaNet [11]. Speed is measured on a NVIDIA 2080Ti GPU. For fair comparison, we only measure the network latency for all detectors. DetOH achieves competitive performance compared with recent methods including anchor-based ones.

Table 1. DetOH vs. Other State-of-the-art Two-stage or One-stage Detectors. DetOH outperforms a few recent anchor-based and anchor-free detectors.

Method	Backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods:							
Faster R-CNN by G-RMI [33]	Inception-ResNet-v2 [28]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN + + + [28]	ResNet-101	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w/ FPN [23]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN w/ TDM [14]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	52.1
One-stage methods:							
YOLOv2 [5]	DarkNet-19	21.6	44.0	19.2	5.0	22.4	35.5
SSD [4]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
YOLOv3 [6]	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9
DSSD [15]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1

(*continued*)

Table 1. (*continued*)

Method	Backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
RetinaNet [11]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
CornerNet [16]	Hourglass-104	40.5	56.5	43.1	19.4	42.7	53.9
FSAF [19]	ResNeXt-101-FPN	42.9	63.8	46.3	26.6	46.2	52.7
FCOS [9]	ResNet-101-FPN	43.2	62.4	46.8	26.1	46.2	52.8
CenterNet [8]	Hourglass-104	44.9	62.4	48.1	25.6	47.4	**57.4**
DetOH	Hourglass-104	44.7	63.1	48.1	28.3	48.4	54.7
DetOH	ResNet-101-BiFPN	**46.1**	**65.3**	**49.8**	**29.2**	**49.6**	56.5

4.3 Application

In the multi-source sensor intelligent security system, we apply the DetOH model to detect objects in the image stream. We apply the trained model to the digital processing chip Hi3519AV100 and successfully achieve object detection, proving a good balance of speed and accuracy. Although Hi3519AV100 has 1.7 TOPS neural network compu-ting performance, it supports the Caffe framework only and is based on Caffe-1.0. It does not support the attention mechanism and channel shuffling, which makes most of the recent mobile models difficult to apply. Considering the huge data processing tasks with only 1 GB memory, it is unfriendly to large models.

Our model runs on the chip's AI-accelerated unit, while post-processing can run on CPU. This is another advantage of DetOH, which can handle different detection processes with different arithmetic units. We used pipeline acceleration for model inference and post-processing. For a single frame size of 1920 $\times$ 1080 pixels, the processing time can be reduced to 380 ms.

5 Ablation Study

In this section, we conduct an ablation study on the three main ideas in this work. The first is to compare different heatmaps. The effects of different loss functions on the results are then compared. Finally, we look at the difference that boundary fine-tuning brings.

We adjusted the dataset for faster ablation studies. The COCO dataset was too large and the training time was too long, so we replaced it with a small private dataset. The training set of this dataset contains 3,518 images, and the test set contains 704 images, all of which are of 640 $\times$ 360 pixels. And the data set covers only three categories: person, vehicle, and aircraft. At the same time, we used a smaller backbone network, MobileUnet [32]. It has fewer parameters and is easier to converge. This means that the training time per ablation experiment can be reduced.

5.1 Heatmaps

This section will introduce the influence of heatmap hyperparameters on the object detection, i.e., the degree of polynomial and the attenuation coefficient in Eq. (1). To see the change of its probability heat value more conveniently, the change curve of the probability heat value with coordinates is plotted. Figure 5 shows the probability heat value curves with different hyperparameters.

From the derivative at the CP, it is not derivable when the polynomial degree is 1 or 0.5, and its value is 0 when the polynomial degree is 2. Our experiments show that neural networks fit a derivable function more easily. To evaluate the impact of heatmaps generated by different hyperparameter combinations on object detection accuracy, an ablation study was conducted. The statistical results of the influence of different hyperparameter combinations on accuracy are shown in Table 2.

From the degree of polynomials, the best performer is the quadratic elliptic heatmaps, followed by the linear diamond heatmaps, and the worst is the square star heatmaps. It can be seen that CNNs are better at fitting derivable convex function graphs. When the probability heat value does not fall smoothly at the CP, it is difficult to simulate the effect of this mutation.

From the attenuation coefficient, when it is large, the heat spot area will be more concentrated near the center of the object, and the accuracy rate is higher. When it is small, the heat spots are more dispersed and the accuracy rate decreases. This is because there is overlap between multiple objects of the same kind. When the two object areas partially overlap, the scattered hot spots will also be more likely to intersect. Although the probability heat value of the intersection area to the maximum value can reduce the influence of the CP position offset, it still brings trouble to the calculation of size parameters. If the two targets are too close together, the size calculation will be on the larger side.

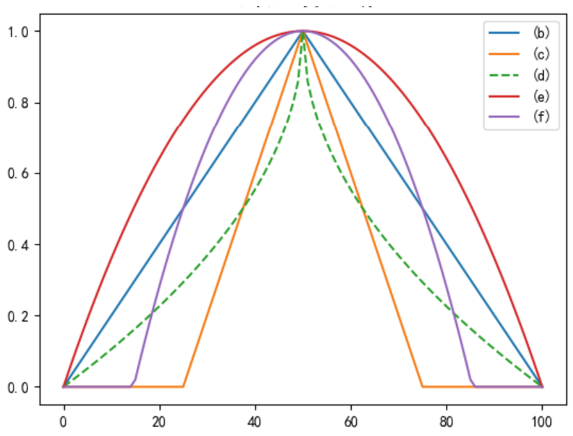

Fig. 5. Curve of probability heat value as a function of abscissa. This is a target with a width of 101 and the ordinate is on the central axis. (b) (c) (d) (e) (f) corresponds to that in Fig. 3, respectively.

Table 2. Quantitative results from Different Hyperparameter Combinations.

	Degree of polynomial	Attenuation coefficient	AP	AP_{50}	AP_{75}
(b)	$r = 1$	$\alpha = 0.5$	66.6	96.0	73.9
(c)	$r = 1$	$\alpha = 1$	72.5	**98.3**	84.3
(d)	$r = 0.5$	$\alpha = 0.5$	52.2	94.1	49.2
(e)	$r = 2$	$\alpha = 0.5$	72.3	93.7	86.8
(f)	$r = 2$	$\alpha = 1$	**74.4**	97.5	**92.7**

5.2 Loss Function

The loss function adopted in this work is a combined loss function, which is replaced by a different loss function to study its effect on the result. The statistical results of the influence of different loss functions on accuracy are shown in Table 3.

Table 3. Quantitative Results from Different Loss Functions.

Loss Function	AP	AP_{50}	AP_{75}
CE loss	28.0	71.1	22.3
Dice loss	0.483	2.45	0.121
Focal loss	23.7	61.5	24.4
MSE loss	56.0	94.7	45.0
Focal loss + CP loss	44.3	82.3	39.6
CE loss + CP loss	55.3	91.8	58.6
MSE loss + CP loss	**66.6**	**96.0**	**73.9**

By comparing the MSE loss + CP loss with the MSE loss alone, the improvement of the CP loss function is huge. The AP as the main evaluation indicator is improved by 18.9%. In AP_{75}, it has been improved by as much as 64.2%. This shows a huge improvement in size calculations by fine-tuning the position of the CP. But in AP_{50}, it brought only 1.4% improvement. This is because the AP with a large IoU threshold is much more sensitive to the size accuracy than the AP with a small IoU threshold.

Comparing the CE loss + CP loss with the performance using the CE loss alone, it can be found that this improvement is more pronounced. It improved by 97.5% in AP and 162.8% in AP_{75}. In AP_{50}, the addition of the CP loss function brought a 29.1% boost. This shows that the CP loss function can significantly improve the accuracy of size calculation, making the prediction box closer to the bbox then improving the IoU.

5.3 Boundary Fine-Tune

To evaluate the impact of boundary fine-tuning on object detection accuracy, an ablation study was conducted.

Table 4. Quantitative Results from Boundary Fine-tuning.

Boundary Fine-tune	AP	AP_{50}	AP_{75}
	74.4	97.5	92.7
✓	**76.5**	**97.6**	**94.5**

As can be seen from Table 4, boundary fine-tuning brings improvement to AP at high IoU threshold, which illustrates its effectiveness in bounding box fitting. These findings suggest that boundary fine-tuning can play a critical role in improving the accuracy of object detection algorithms.

6 Conclusion

In this work, we propose an anchor-free object detector with only heatmaps (DetOH). Our experiments demonstrate that DetOH is superior to widely used anchor-free object detectors, including CenterNet and FCOS, but with much less model complexity. The single-head design makes it easier to be applied to embedded devices. Now it has been applied to the field of security, making human life more secure. Its network architecture is also suitable for other intensive prediction tasks, such as semantic segmentation. Given its effectiveness and efficiency, we hope it soon be applied to high-level tasks such as multi-object tracking, motion recognition, and behavior understanding.

Acknowledgments. This work was supported inpart by the National Natural Science Foundation of China (61972219), the Research and Development Program of Shenzhen (JCYJ20190813174403598), the Overseas Research Cooperation Fund of Tsinghua Shenzhen International Graduate School (HW2021013), the Guangdong Basic and Applied Basic Research Foundation (2022A1515010417), the Key Project of Shenzhen Municipality (JSGG20211029095545002), the Science and Technology Research Project of Henan Province(222102210096).

References

1. Viola, P., Jones, M.J.: Robust real-time face detection. Int. J. Comput. Vision **57**, 137–154 (2004)
2. Doll´ar, P., Appel, R., Belongie, S., Perona, P.: Fast feature pyramids for object detection. IEEE Trans. Pattern Anal. Mach. Intell. **36**(8), 1532–1545 (2014)
3. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, vol. 28 (2015)
4. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A.C.: Ssd: Single shot multibox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) Computer Vision – ECCV 2016. Lecture Notes in Computer Science, vol. 9905, pp. 21–37. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46448-0_2

5. Redmon, J., Farhadi, A.: Yolo9000: better, faster, stronger. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7263–7271 (2017)
6. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788 (2016)
7. Zhou, X., Wang, D., Krähenbühl, P.: Objects as points. arXiv preprint: arXiv:1904.07850 (2019)
8. Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., Tian, Q.: CenterNet: keypoint triplets for object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 6569–6578 (2019)
9. Tian, Z., Shen, C., Chen, H., He, T.: FCOS: fully convolutional one-stage object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9627–9636 (2019)
10. Tian, Z., Shen, C., Chen, H., He, T.: FCOS: a simple and strong anchorfree object detector. IEEE Trans. Pattern Anal. Mach. Intell. **44**(4), 1922–1933 (2020)
11. Lin, T.Y., Goyal, P., Girshick, R., He, K., Doll´ar, P.: Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2980–2988 (2017)
12. Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. Lecture Notes in Computer Science, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
13. Girshick, R.: Fast R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448 (2015)
14. Shrivastava, A., Sukthankar, R., Malik, J., Gupta, A.: Beyond skip connections: top-down modulation for object detection. arXiv preprint: arXiv:1612.06851 (2016)
15. Fu, C.Y., Liu, W., Ranga, A., Tyagi, A., Berg, A.C.: DSSD: deconvolutional single shot detector. arXiv preprint: arXiv:1701.06659 (2017)
16. Law, H., Deng, J.: CornerNet: detecting objects as paired keypoints. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) Computer Vision –. Lecture Notes in Computer Science, vol. 11218, pp. 765–781. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01264-9_45
17. Yu, J., Jiang, Y., Wang, Z., Cao, Z., Huang, T.: Unitbox: an advanced object detection network. In: Proceedings of the 24th ACM International Conference on Multimedia, pp. 516–520 (2016)
18. Huang, L., Yang, Y., Deng, Y., Yu, Y.: DenseBox: unifying landmark localization with end to end object detection. arXiv preprint: arXiv:1509.04874 (2015)
19. Zhu, C., He, Y., Savvides, M.: Feature selective anchor-free module for single-shot object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 840–849 (2019)
20. Yang, Z., Liu, S., Hu, H., Wang, L., Lin, S.: RepPoints: point set representation for object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9657–9666 (2019)
21. Duan, K., Xie, L., Qi, H., Bai, S., Huang, Q., Tian, Q.: Corner proposal network for anchor-free, two-stage object detection. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) Computer Vision – ECCV 2020. Lecture Notes in Computer Science, vol. 12348, pp. 399–416. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58580-8_24
22. Samet, N., Hicsonmez, S., Akbas, E.: Houghnet: Integrating near and long-range evidence for bottom-up object detection. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) Computer Vision – ECCV 2020. Lecture Notes in Computer Science, vol. 12370, pp. 406–423. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58595-2_25

23. Lin, T.Y., Doll´ar, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2117–2125 (2017)
24. Tan, M., Pang, R., Le, Q.V.: EfficientDet: scalable and efficient object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10781–10790 (2020)
25. Qiu, H., Ma, Y., Li, Z., Liu, S., Sun, J.: Borderdet: Border feature for dense object detection. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) Computer Vision – ECCV 2020. Lecture Notes in Computer Science, vol. 12346, pp. 549–564. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58452-8_32
26. Li, X., et al.: Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection. In: Advances in Neural Information Processing Systems, vol. 33, pp. 21002–21012 (2020)
27. Dai, X., et al.: Dynamic head: Unifying object detection heads with attentions. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7373–7382 (2021)
28. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.: Inception-v4, inceptionresnet and the impact of residual connections on learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31 (2017)
29. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
30. Yu, F., Wang, D., Shelhamer, E., Darrell, T.: Deep layer aggregation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2403–2412 (2018)
31. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) Computer Vision – ECCV 2016. Lecture Notes in Computer Science, vol. 9912, pp. 483–499. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46484-8_29
32. Abdollahi, A., Pradhan, B., Alamri, A.: VNet: an end-to-end fully convolutional neural network for road extraction from high-resolution remote sensing data. IEEE Access **8**, 179424–179436 (2020)
33. Jing, J., Wang, Z., R¨atsch, M., Zhang, H.: Mobile-Unet: an efficient convolutional neural network for fabric defect detection. Text. Res. J. **92**(1–2), 30–42 (2022)
34. Huang, J., et al.: Speed/accuracy trade-offs for modern convolutional object detectors. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7310–7311 (2017)