# Oriented R-CNN for Object Detection

Xingxing Xie Gong Cheng* Jiabao Wang Xiwen Yao Junwei Han
School of Automation, Northwestern Ploytechnical University, Xi'an, China
{xiexing,jbwang}@mail.nwpu.edu.cn {gcheng,yaoxiwen,jhan}@nwpu.edu.cn

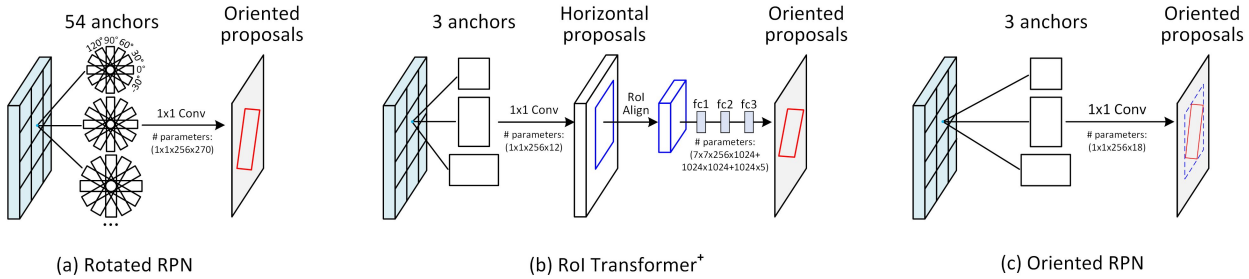


Figure 1: Comparisons of different schemes for generating oriented proposals. (a) Rotated RPN densely places rotated anchors with different scales, ratios, and angles. (b) RoI Transformer+ learns oriented proposal from horizontal RoI. It involvs RPN, RoI Alignment, and regression. (c) Our proposed oriented RPN generates high-quality proposals in a nearly cost-free manner. The number of parameters of oriented RPN is about 1/3000 of RoI Transformer+ and 1/15 of rotated RPN.

## Abstract

*Current state-of-the-art two-stage detectors generate oriented proposals through time-consuming schemes. This diminishes the detectors' speed, thereby becoming the computational bottleneck in advanced oriented object detection systems. This work proposes an effective and simple oriented object detection framework, termed Oriented R-CNN, which is a general two-stage oriented detector with promising accuracy and efficiency. To be specific, in the first stage, we propose an oriented Region Proposal Network (oriented RPN) that directly generates high-quality oriented proposals in a nearly cost-free manner. The second stage is oriented R-CNN head for refining oriented Regions of Interest (oriented RoIs) and recognizing them. Without tricks, oriented R-CNN with ResNet50 achieves state-of-the-art detection accuracy on two commonly-used datasets for oriented object detection including DOTA (75.87% mAP) and HRSC2016 (96.50% mAP), while having a speed of 15.1 FPS with the image size of 1024×1024 on a single RTX 2080Ti. We hope our work could inspire rethinking the design of oriented detectors and serve as a baseline for oriented object detection. Code is available at* `https://github.com/jbwang1997/OBBDetection`.

*Corresponding author. + denotes the official version of RoI Transformer implemented on AerialDetection (the same below).

## 1. Introduction

Most existing state-of-the-art oriented object detection methods [7, 37, 41] depend on proposal-driven frameworks, like Fast/Faster R-CNN [11, 10, 32]. They involve two key steps: (i) generating oriented proposals and (ii) refining proposals and classifying them into different categories. Nevertheless, the step of producing oriented proposals is computationally expensive.

One of the early methods of generating oriented proposals is rotated Region Proposal Network (rotated RPN for short) [26], which places 54 anchors with different angles, scales and aspect rations (3 scales×3 ratios×6 angles) on each location, as shown in Figure 1(a). The introduction of rotated anchors improves the recall and demonstrates good performance when the oriented objects distribute sparsely. However, abundant anchors cause massive computation and memory footprint. To address this issue, RoI Transformer [7] learns oriented proposals from horizontal RoIs by complex process, which involves RPN, RoI Alignment and regression (see Figure 1(b)). The RoI Transformer scheme provides promising oriented proposals and drastically reduces the number of rotated anchors, but also brings about expensive computation cost. Now, how to design an elegant and efficient solution to generate oriented proposals is the key to breaking the computational bottleneck in state-of-the-art oriented detectors.

To push the envelope further: we investigate why the efficiency of region proposal-based oriented detectors has trailed thus far. Our observation is that the main obstacle impeding the speed of proposal-driven detectors is from the stage of proposal generation. A natural and intuitive question to ask is: can we design a general and simple oriented region proposal network (oriented RPN for short) to generate high-quality oriented proposals directly? Motivated by this question, this paper presents a simple two-stage oriented object detection framework, called oriented R-CNN, which obtains state-of-the-art detection accuracy, while keeping competitive efficiency in comparison with one-stage oriented detectors.

To be specific, in the first stage of oriented R-CNN, a conceptually simple oriented RPN is presented (see Figure 1(c)). Our oriented RPN is a kind of light-weight fully convolutional network, which has extremely fewer parameters than rotated RPN and RoI transformer+, thus reducing the risk of overfitting. We realize it by changing the number of output parameters of RPN regression branch from four to six. There is no such thing as a free lunch. The design of oriented RPN benefits from our proposed representation scheme of oriented objects, named midpoint offset representation. For arbitrary-oriented objects in images, we utilize six parameters to represent them. The midpoint offset representation inherits horizontal regression mechanism, as well as provides bounded constraints for predicting oriented proposals. The second stage of oriented R-CNN is oriented R-CNN detection head: extracting the features of each oriented proposal by rotated RoI alignment and performing classification and regression.

Without bells and whistles, we evaluate our oriented R-CNN on two popular benchmarks for oriented object detection, namely DOTA and HRSC2016. Our method with ResNet-50-FPN surpasses the accuracy of all existing state-of-the-art detectors, achieving 75.87% mAP on the DOTA dataset and 96.50% mAP on the HRSC2016 dataset, while running at 15.1 FPS with the image size of $1024\times1024$ on a single RTX 2080Ti. Thus, the oriented R-CNN is a practical object detection framework in terms of both accuracy and efficiency. We hope our method will inspire rethinking the design of oriented object detectors and oriented object regression scheme, as well as serve as a solid baseline for oriented object detection. For future research, our code is available at `https://github.com/jbwang1997/OBBDetection`.

## 2. Related Work

In the past decade, remarkable progresses [23, 33, 2, 42, 31, 24, 45, 13, 20, 29] have been made in the field of object detection. As an extended branch of object detection, oriented object detection [7, 37, 26, 12, 5, 28] has received extensive attention driven by its wide applications.

General object detection methods (e.g., Faster R-CNN) relying on horizontal bounding boxes could not tightly locate oriented objects in images, because a horizontal bounding box may contain more background regions or multiple objects. This results in the inconsistency between the final classification confidence and localization accuracy. To address this issue, much attention has been devoted. For instance, Xia et al. [36] built a large-scale object detection benchmark with oriented annotations, named DOTA. Many existing oriented object detectors [7, 37, 41, 26, 25, 18] are mainly based on typical proposal-based object detection frameworks. A nature solution to oriented object is to set rotated anchors [26, 25], such as rotated RPN [26], in which the anchors with different angles, scales and aspect ratios are placed on each location. These densely rotated anchors lead to extensive computations and memory footprint.

To decrease the large number of rotated anchors and reduce the mismatch between features and objects, Ding et.al [7] proposed the RoI transformer that learns rotated RoIs from horizontal RoIs produced by RPN. This manner greatly boosts the detection accuracy of oriented objects. However, it makes the network heavy and complex because it involves fully-connected layers and RoI alignment operation during the learning of rotated RoIs. To address the challenges of small, cluttered, and rotated object detection, Yang et.al [41] built an oriented object detection method on the generic object detection framework of Faster R-CNN. Xu et.al [37] proposed a new oriented object representation, termed gliding vertexes. It achieves oriented object detection by learning four vertex gliding offsets on the regression branch of Faster R-CNN head. However, these two methods both adopt horizontal RoIs to perform classification and oriented bounding box regression. They still suffer from severe misalignment between objects and features.

In addition, some works [12, 28, 39, 27, 43, 16, 40, 47, 15] have explored one-stage or anchor-free oriented object detection frameworks: outputting object classes and oriented bounding boxes without region proposal generation and RoI alignment operation. For example, Yang et.al [39] proposed a refined one-stage oriented object detector, which involves two key improvements, including feature refinement and progressive regression, to address the problem of feature misalignment. Ming et.al [27] designed a new label assignment strategy for one-stage oriented object detection based on RetinaNet [22]. It assigns the positive or negative anchors dynamically through a new matching strategy. Han et.al [12] proposed a single-shot alignment network ($S^2$ANet) for oriented object detection. $S^2$ANet aims at alleviating the inconsistency between the classification score and location accuracy via deep feature alignment. Pan et.al [28] devised a dynamic refinement network (DRN) for oriented object detection based on the anchor-free object detection method CenterNet [46].

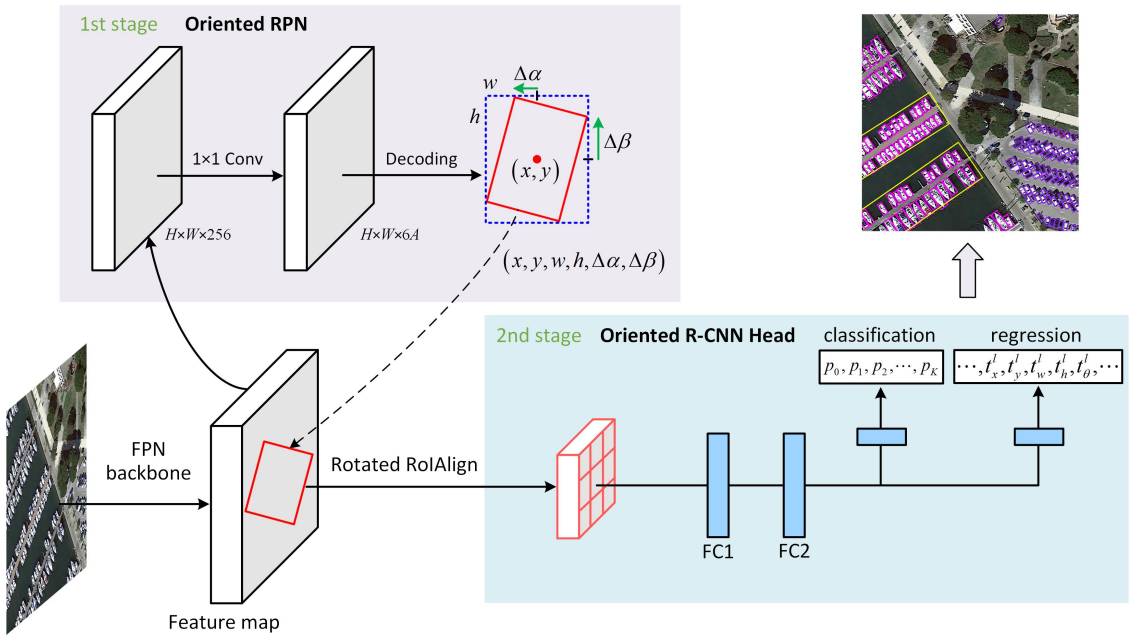


Figure 2: Overall framework of oriented R-CNN, which is a two-stage detector built on FPN. The first stage generates oriented proposals by oriented RPN and the second stage is oriented R-CNN head to classify proposals and refine their spatial locations. For clear illustration, we do not show the FPN as well as the classification branch in oriented RPN.

In contrast with the above methods, our work falls within proposal-based oriented object detection methods. It focuses on designing a high-efficiency oriented RPN to break the computational bottleneck of generating oriented proposals.

## 3. Oriented R-CNN

Our proposed object detection method, called oriented R-CNN, consists of an oriented RPN and an oriented R-CNN head (see Figure 2). It is a two-stage detector, where the first stage generates high-quality oriented proposals in a nearly cost-free manner and the second stage is oriented R-CNN head for proposal classification and regression. Our FPN backbone follows [21], which produces five levels of features $\{P_2, P_3, P_4, P_5, P_6\}$. For simplicity, we do not show the FPN architecture as well as the classification branch in oriented RPN. Next, we describe the oriented RPN and oriented R-CNN head in detail.

### 3.1. Oriented RPN

Given an input image of any size, oriented RPN outputs a sparse set of oriented proposals. The entire process can be modeled by light-weight fully-convolutional networks.

Specifically, it takes five levels of features $\{P_2, P_3, P_4, P_5, P_6\}$ of FPN as input and attaches a head of the same design (a 3×3 convolutional layer and two sibling 1×1 convolutional layers) to each level of features. We assign three horizontal anchors with three aspect rations {1:2, 1:1, 2:1} to each spatial location in all levels of features. The anchors have the pixel areas of $\{32^2, 64^2, 128^2, 256^2, 512^2\}$ on $\{P_2, P_3, P_4, P_5, P_6\}$, respectively. Each anchor $\boldsymbol{a}$ is denoted by a 4-dimensional vector $\boldsymbol{a} = (a_x, a_y, a_w, a_h)$, where $(a_x, a_y)$ is the center coordinate of the anchor, $a_w$ and $a_h$ represent the width and height of the anchor. One of the two sibling 1×1 convolutional layers is regression branch: outputting the offset $\boldsymbol{\delta} = (\delta_x, \delta_y, \delta_w, \delta_h, \delta_\alpha, \delta_\beta)$ of the proposals relative to the anchors. At each location of feature map, we generate $A$ proposals ($A$ is the number of anchors at each location, and it equals to 3 in this work), thus the regression branch has $6A$ outputs. By decoding the regression outputs, we can obtain the oriented proposal. The process of decoding is described as follows:

$$\begin{cases} \Delta\alpha = \delta_\alpha \cdot w, \quad \Delta\beta = \delta_\beta \cdot h \\ w = a_w \cdot e^{\delta_w}, \quad h = a_h \cdot e^{\delta_h} \\ x = \delta_x \cdot a_w + a_x, \quad y = \delta_y \cdot a_h + a_y \end{cases} \tag{1}$$

where $(x, y)$ is the center coordinate of the predicted proposal, $w$ and $h$ are the width and height of the external rectangle box of the predicted oriented proposal. $\Delta\alpha$ and $\Delta\beta$ are the offsets relative to the midpoints of the top and right sides of the external rectangle. Finally, we produce oriented proposals according to $(x, y, w, h, \Delta\alpha, \Delta\beta)$.

The other sibling convolutional layer estimates the objectness score for each oriented proposal. For clear illustration, we omit the scoring branch in Figure 2. The oriented RPN is acctually a natural and intuitive idea, but its key lies

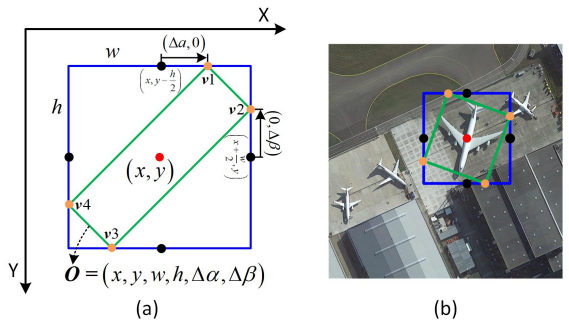


Figure 3: Illustration of midpoint offset representation. (a) The schematic diagram of midpoint offset representation. (b) An example of midpoint offset representation.

in the representation of oriented objects. Under this circumstance, we design a new and simple representation scheme of oriented objects, called midpoint offset representation.

#### 3.1.1 Midpoint Offset Representation

We propose a novel representation scheme of oriented objects, named midpoint offset representation, as shown in Figure 3. The black dots are the midpoints of each side of the horizontal box, which is the external rectangle of the oriented bounding box $\boldsymbol{O}$. The orange dots stand for the vertexes of the oriented bounding box $\boldsymbol{O}$.

Specifically, we use an oriented bounding box $\boldsymbol{O}$ with six parameters $\boldsymbol{O} = (x, y, w, h, \Delta\alpha, \Delta\beta)$ to represent an object computed by Equation (1). Through the six parameters, we can obtain the coordinate set $\mathbf{v} = (\boldsymbol{v}1, \boldsymbol{v}2, \boldsymbol{v}3, \boldsymbol{v}4)$ of four vertexes for each proposal. Here, $\Delta\alpha$ is the offset of $\boldsymbol{v}1$ with respect to the midpoint $(x, y - h/2)$ of the top side of the horizontal box. According to the symmetry, $-\Delta\alpha$ represents the offset of $\boldsymbol{v}3$ with respect to the bottom midpoint $(x, y + h/2)$. $\Delta\beta$ stands for the offset of $\boldsymbol{v}2$ with respect to the right midpoint $(x + w/2, y)$, and $-\Delta\beta$ is the offset of $\boldsymbol{v}4$ with respect to the left midpoint $(x - w/2, y)$. Thus, the coordinates of four vertexes can be expressed as follows.

$$\begin{cases} \boldsymbol{v}1 = (x, y - h/2) + (\Delta\alpha, 0) \\ \boldsymbol{v}2 = (x + w/2, y) + (0, \Delta\beta) \\ \boldsymbol{v}3 = (x, y + h/2) + (-\Delta\alpha, 0) \\ \boldsymbol{v}4 = (x - w/2, y) + (0, -\Delta\beta) \end{cases} \tag{2}$$

With the representation manner, we implement the regression for each oriented proposal through predicting the parameters $(x, y, w, h)$ for its external rectangle and inferring the parameters $(\Delta\alpha, \Delta\beta)$ for its midpoint offset.

#### 3.1.2 Loss Function

To train oriented RPN, the positive and negative samples are defined as follows. First, we assign a binary label $p^* \in \{0, 1\}$ to each anchor. Here, 0 and 1 mean that the anchor belongs to positive or negative sample. To be specific,

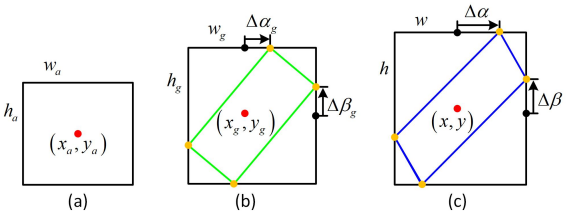


Figure 4: The illustration of box-regression parameterization. Black dots are the midpoints of the top and right sides, and orange dots are the vertexes of the oriented bounding box. (a) Anchor. (b) Ground-truth box. (c) Predicted box.

we consider an anchor as positive sample under one of the two conditions: (i) an anchor having an Intersection-over-Union (IoU) overlap higher than 0.7 with any ground-truth box, (ii) an anchor having the highest IoU overlap with a ground-truth box and the IoU is higher than 0.3. The anchors are labeled as negative samples when their IoUs are lower than 0.3 with ground-truth box. The anchors that are neither positive nor negative are considered as invalid samples, which are ingnored in the training process. It is worth noting that the above-mentioned ground-truth boxes refer to the external rectangles of oriented bounding boxes.

Next, we define the loss function $L_1$ as follows:

$$L_1 = \frac{1}{N}\sum_{i=1}^{N} F_{cls}\left(p_i, p_i^*\right) + \frac{1}{N} p_i^* \sum_{i=1}^{N} F_{reg}\left(\boldsymbol{\delta}_i, \boldsymbol{t}_i^*\right) \tag{3}$$

Here, $i$ is the index of the anchors and $N$ (by default $N$=256) is the total number of samples in a mini-batch. $p_i^*$ is the ground-truth label of the $i$-th anchor. $p_i$ is the output of the classification branch of oriented RPN, which denotes the probability that the proposal belongs to the foreground. $\boldsymbol{t}_i^*$ is the supervision offset of the ground-truth box relative to $i$-th anchor, which is a parameterized 6-dimensional vector $\boldsymbol{t}_i^* = \left(t_x^*, t_y^*, t_w^*, t_h^*, t_\alpha^*, t_\beta^*\right)$ from the regression branch of oriented RPN, denoting the offset of the predicted proposal relative to the $i$-th anchor. $F_{cls}$ is the cross entropy loss. $F_{reg}$ is the Smooth $L1$ loss. For box regression (see Figure 4), we adopt the affine transformation, which is formulated as follows:

$$\begin{cases} \delta_\alpha = \Delta\alpha / w, & \delta_\beta = \Delta\beta / h \\ \delta_w = \log\left(w / w_a\right), & \delta_h = \log\left(h / h_a\right) \\ \delta_x = \left(x - x_a\right) / w_a, & \delta_y = \left(y - y_a\right) / h_a \\ t_\alpha^* = \Delta\alpha_g / w_g, & t_\beta^* = \Delta\beta_g / h_g \\ t_w^* = \log\left(w_g / w_a\right), & t_h^* = \log\left(h_g / h_a\right) \\ t_x^* = \left(x_g - x_a\right) / w_a, & t_y^* = \left(x_g - x_a\right) / h_a \end{cases} \tag{4}$$

where $(x_g, y_g)$, $w_g$ and $h_g$ are the center coordinate, width, and height of external rectangle, respectively. $\Delta\alpha_g$ and $\Delta\beta_g$ are the offsets of the top and right vertexes relative to the midpoints of top and left sides.

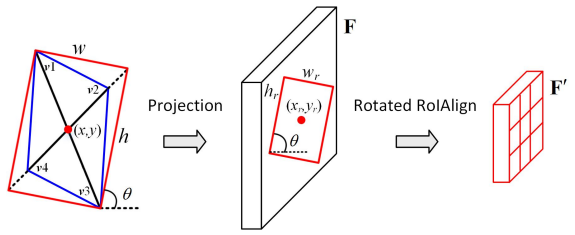


Figure 5: Illustration of the process of rotated RoIAlign. Blue box is a parallelogram proposal generated by oriented RPN, and the most-left red box is its corresponding rectangular proposal used for projection and rotated RoIAlign.

### 3.2. Oriented R-CNN Head

Oriented R-CNN head takes the feature maps $\{P_2, P_3, P_4, P_5\}$ and a set of oriented proposals as input. For each oriented proposal, we use rotated RoI alignment (rotated RoIAlign for short) to extract a fixed-size feature vector from its corresponding feature map. Each feature vector is fed into two fully-connected layers (FC1 and FC2, see Figure 2), followed by two sibling fully-connected layers: one outputs the probability that the proposal belongs to $K$+1 classes ($K$ object classes plus 1 background class) and the other one produces the offsets of the proposal for each of the $K$ object classes.

#### 3.2.1 Rotated RoIAlign

Rotated RoIAlign is an operation for extracting rotation-invariant features from each oriented proposal. Now, we describe the process of rotated RoIAlign according to Figure 5. The oriented proposal generated by oriented RPN is usually a parallelogram (blue box in Figure 5), which is denoted with the parameters $\mathbf{v} = (\boldsymbol{v}1, \boldsymbol{v}2, \boldsymbol{v}3, \boldsymbol{v}4)$, where $\boldsymbol{v}1$, $\boldsymbol{v}2$, $\boldsymbol{v}3$, and $\boldsymbol{v}4$ are its vertex coordinates. For ease of computing, we need to adjust each parallelogram to a rectangular with direction. To be specific, we achieve this by extending the shorter diagonal (the line from $\boldsymbol{v}2$ to $\boldsymbol{v}4$ in Figure 5) of the parallelogram to have the same length as the longer diagonal. After this simple operation, we obtain the oriented rectangular $(x, y, w, h, \theta)$ (red box in Figure 5) from the parallelogram, where $\theta \in [-\pi/2, \pi/2]$ is defined by the intersection angle between the horizontal axis and the longer side of the rectangular.

We next project the oriented rectangular $(x, y, w, h, \theta)$ to the feature map $\mathbf{F}$ with the stride of $s$ to obtain a rotated RoI, which is defined by $(x_r, y_r, w_r, h_r, \theta)$ through the following operation.

$$\begin{cases} w_r = w/s, & h_r = h/s \\ x_r = \lfloor x/s \rfloor, & y_r = \lfloor y/s \rfloor \end{cases} \tag{5}$$

Then, each rotated RoI is divided into $m \times m$ grids ($m$ defaults to 7) to get a fixed-size feature map $\mathbf{F}'$ with the dimension of $m \times m \times C$. For each grid with index $(i, j)$ $(0 \le i, j \le m-1)$ in the $c$-th channel $(1 \le c < C)$, its value is calculated as follows:

$$\mathbf{F}'_c(i,j) = \sum_{(x,y)\in \text{area}(i,j)} \mathbf{F}_c(R(x,y,\theta))/n \tag{6}$$

where $\mathbf{F}_c$ is the feature of the $c$-th channel, $n$ is the number of samples localized within each grid, and $\text{area}(i, j)$ is the coordinate set contained in the grid with index $(i, j)$. $R(\cdot)$ is a rotation transformation the same as [7].

### 3.3. Implementation Details

Oriented R-CNN is trained in an end-to-end manner by jointly optimizing oriented RPN and oriented R-CNN head. During inference, the oriented proposals generated by oriented RPN generally have high overlaps. In order to reduce the redundancy, we remain 2000 proposals per FPN level in the first stage, followed by Non-Maximum Suppression (NMS). Considering the inference speed, the horizontal NMS with the IoU threshold of 0.8 is adopted. We merge the remaining proposals from all levels, and choose top-1000 ones based on their classification scores as the input of the second stage. In the second stage, ploy NMS for each object class is performed on those predicted oriented bounding boxes whose class probability is higher than 0.05. The ploy NMS IoU threshold is 0.1.

## 4. Experiments

To evaluate our proposed method, we conduct extensive experiments on two most widely-used oriented object detection datasets, namely DOTA [36] and HRSC2016 [25].

### 4.1. Datasets

DOTA is a large-scale dataset for oriented object detection. It contains 2806 images and 188282 instances with oriented bounding box annotations, covered by the following 15 object classes: Bridge (BR), Harbor (HA), Ship (SH), Plane (PL), Helicopter (HC), Small vehicle (SV), Large vehicle (LV), Baseball diamond (BD), Ground track field (GTF), Tennis court (TC), Basketball court (BC), Soccer-ball field (SBF), Roundabout (RA), Swimming pool (SP), and Storage tank (ST). The image size of the DOTA dataset is large: from 800×800 to 4000×4000 pixels. We use training and validation sets for training and the rest for testing. The detection accuracy is obtained by submitting testing results to DOTA's evaluation server.

HRSC2016 is another widely-used dataset for arbitrary-oriented ship detection. It contains 1061 images with the size ranging from 300×300 to 1500×900. Both the training set (436 images) and validation set (181 images) are used for training and the remaining for testing. For the detection accuracy on the HRSC2016, we adopt the mean average precision (mAP) as evaluation criteria, which is consistent with PASCAL VOC 2007 [8] and VOC 2012.

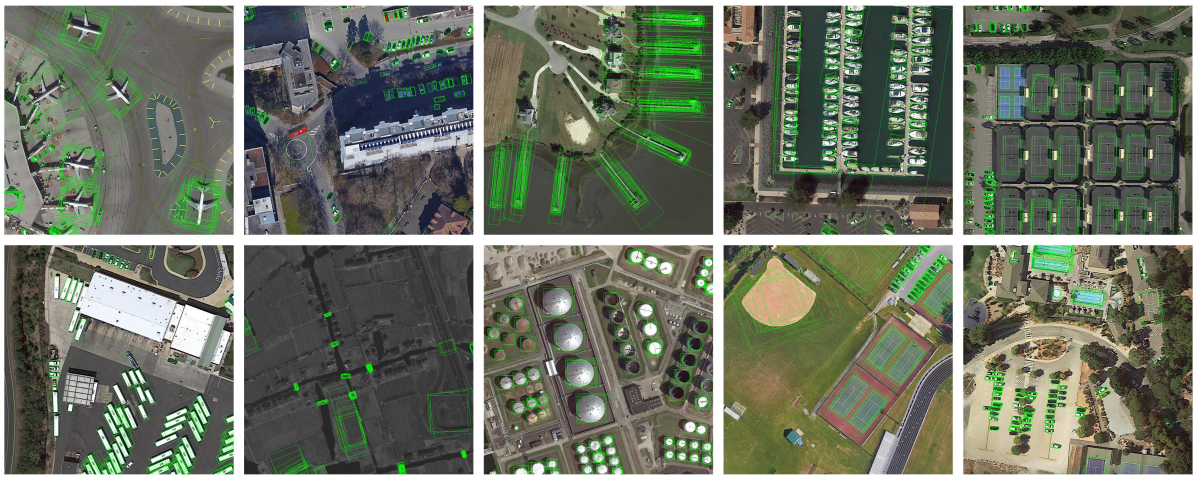

Figure 6: Proposals generated by oriented RPN on the DOTA dataset. The top-200 proposals per image are displayed.

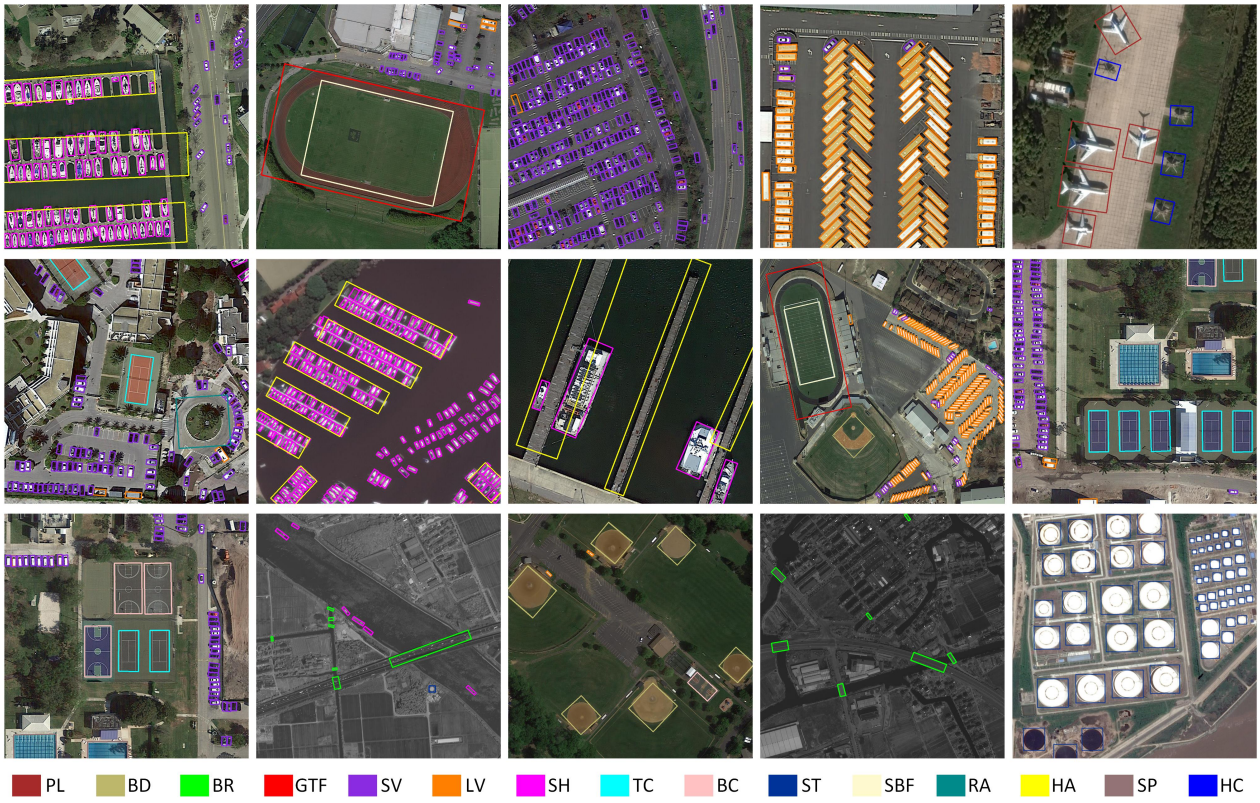


Figure 7: Examples of detection results on the DOTA dataset using oriented R-CNN with R-50-FPN backbone. The confidence threshold is set to 0.3 when visualizing these results. One color stands for one object class.

| Method | $R_{300}$ | $R_{1000}$ | $R_{2000}$ |
|---|---|---|---|
| Oriented RPN | 81.60 | 92.20 | **92.80** |

Table 1: Recall results on the DOTA validation set.

### 4.2. Parameter settings

We use a single RTX 2080Ti with the batch size of 2 for training. The inference time is also tested on a single RTX 2080Ti. The experimental results are produced on the mmdetection platform [3]. ResNet50 [14] and ResNet101 [14] are used as our backbones. They are pre-trained on ImageNet [6]. Horizontal and vertical flipping are adopted as data augmentation during training. We optimize the overall network with SGD algorithm with the momentum of 0.9 and the weight decay of 0.0001. On the

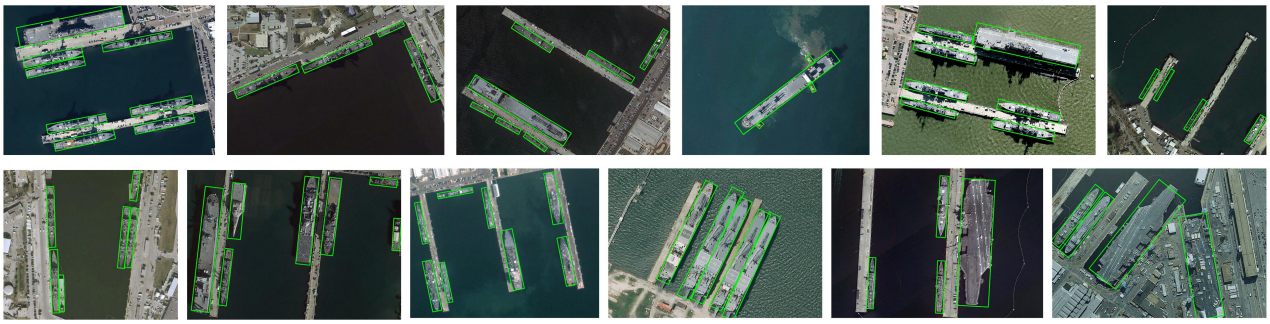

Figure 8: Examples of detection results on the HRSC2016 dataset using oriented R-CNN with R-50-FPN backbone. The oriented bounding boxes whose scores are higher than 0.3 are shown.

| Method | Backbone | PL | BD | BR | GTF | SV | LV | SH | TC | BC | ST | SBF | RA | HA | SP | HC | mAP |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **One-stage** | | | | | | | | | | | | | | | | | |
| RetinaNet-O† | R-50-FPN | 88.67 | 77.62 | 41.81 | 58.17 | 74.58 | 71.64 | 79.11 | 90.29 | 82.18 | 74.32 | 54.75 | 60.60 | 62.57 | 69.67 | 60.64 | 68.43 |
| DRN [28] | H-104 | 88.91 | 80.22 | 43.52 | 63.35 | 73.48 | 70.69 | 84.94 | 90.14 | 83.85 | 84.11 | 50.12 | 58.41 | 67.62 | 68.60 | 52.50 | 70.70 |
| R3Det [39] | R-101-FPN | 88.76 | 83.09 | 50.91 | 67.27 | 76.23 | 80.39 | 86.72 | 90.78 | 84.68 | 83.24 | 61.98 | 61.35 | 66.91 | 70.63 | 53.94 | 73.79 |
| PIoU [4] | DLA-34 | 80.90 | 69.70 | 24.10 | 60.20 | 38.30 | 64.40 | 64.80 | 90.90 | 77.20 | 70.40 | 46.50 | 37.10 | 57.10 | 61.90 | 64.00 | 60.50 |
| RSDet [30] | R-101-FPN | 89.80 | 82.90 | 48.60 | 65.20 | 69.50 | 70.10 | 70.20 | 90.50 | 85.60 | 83.40 | 62.50 | 63.90 | 65.60 | 67.20 | 68.00 | 72.20 |
| DAL [27] | R-50-FPN | 88.68 | 76.55 | 45.08 | 66.80 | 67.00 | 76.76 | 79.74 | 90.84 | 79.54 | 78.45 | 57.71 | 62.27 | 69.05 | 73.14 | 60.11 | 71.44 |
| $S^2$ANet [12] | R-50-FPN | 89.11 | 82.84 | 48.37 | 71.11 | 78.11 | 78.39 | 87.25 | 90.83 | 84.90 | 85.64 | 60.36 | 62.60 | 65.26 | 69.13 | 57.94 | 74.12 |
| ***two-stage*** | | | | | | | | | | | | | | | | | |
| ICN [1] | R-101-FPN | 81.36 | 74.30 | 47.70 | 70.32 | 64.89 | 67.82 | 69.98 | 90.76 | 79.06 | 78.20 | 53.64 | 62.90 | 67.02 | 64.17 | 50.23 | 68.16 |
| Faster R-CNN-O† | R-50-FPN | 88.44 | 73.06 | 44.86 | 59.09 | 73.25 | 71.49 | 77.11 | 90.84 | 78.94 | 83.90 | 48.59 | 62.95 | 62.18 | 64.91 | 56.18 | 69.05 |
| CAD-Net [44] | R-101-FPN | 87.80 | 82.40 | 49.40 | 73.50 | 71.10 | 63.50 | 76.60 | 90.90 | 79.20 | 73.30 | 48.40 | 60.90 | 62.00 | 67.00 | 62.20 | 69.90 |
| RoI Transformer [7] | R-101-FPN | 88.64 | 78.52 | 43.44 | 75.92 | 68.81 | 73.68 | 83.59 | 90.74 | 77.27 | 81.46 | 58.39 | 53.54 | 62.83 | 58.93 | 47.67 | 69.56 |
| SCRDet [41] | R-101-FPN | 89.98 | 80.65 | 52.09 | 68.36 | 68.36 | 60.32 | 72.41 | 90.85 | **87.94** | 86.86 | 65.02 | 66.68 | 66.25 | 68.24 | 65.21 | 72.61 |
| RoI Transformer+ | R-50-FPN | 88.65 | 82.60 | 52.53 | 70.87 | 77.93 | 76.67 | 86.87 | 90.71 | 83.83 | 82.51 | 53.95 | 67.61 | 74.67 | 68.75 | 61.03 | 74.61 |
| Gliding Vertex [37] | R-101-FPN | 89.64 | 85.00 | 52.26 | 77.34 | 73.01 | 73.14 | 86.82 | 90.74 | 79.02 | 86.81 | 59.55 | 70.91 | 72.94 | 70.86 | 57.32 | 75.02 |
| FAOD [19] | R-101-FPN | 90.21 | 79.58 | 45.49 | 76.41 | 73.18 | 68.27 | 79.56 | 90.83 | 83.40 | 84.68 | 53.40 | 65.42 | 74.17 | 69.69 | 64.86 | 73.28 |
| CenterMap-Net [35] | R-50-FPN | 88.88 | 81.24 | 53.15 | 60.65 | 78.62 | 66.55 | 78.10 | 88.83 | 77.80 | 83.61 | 49.36 | 66.19 | 72.10 | 72.36 | 58.70 | 71.74 |
| FR-Est [9] | R-101-FPN | 89.63 | 81.17 | 50.44 | 70.19 | 73.52 | 77.98 | 86.44 | 90.82 | 84.13 | 83.56 | 60.64 | 66.59 | 70.59 | 66.72 | 60.55 | 74.20 |
| Mask OBB [34] | R-50-FPN | 89.61 | 85.09 | 51.85 | 72.90 | 75.28 | 73.23 | 85.57 | 90.37 | 82.08 | 85.05 | 55.73 | 68.39 | 71.61 | 69.87 | 66.33 | 74.86 |
| ***Ours*** | | | | | | | | | | | | | | | | | |
| Oriented R-CNN | R-50-FPN | 89.46 | 82.12 | 54.78 | 70.86 | 78.93 | 83.00 | 88.20 | 90.90 | 87.50 | 84.68 | 63.97 | 67.69 | 74.94 | 68.84 | 52.28 | 75.87 |
| Oriented R-CNN | R-101-FPN | 88.86 | 83.48 | 55.27 | 76.92 | 74.27 | 82.10 | 87.52 | **90.90** | 85.56 | 85.33 | 65.51 | 66.82 | 74.36 | 70.15 | 57.28 | 76.28 |
| Oriented R-CNN‡ | R-50-FPN | 89.84 | **85.43** | 61.09 | 79.82 | **79.71** | **85.35** | **88.82** | 90.88 | 86.68 | 87.73 | 72.21 | **70.80** | 82.42 | 78.18 | **74.11** | **80.87** |
| Oriented R-CNN‡ | R-101-FPN | **90.26** | 84.74 | **62.01** | **80.42** | 79.04 | 85.07 | 88.52 | 90.85 | 87.24 | **87.96** | **72.26** | 70.03 | **82.93** | **78.46** | 68.05 | 80.52 |

Table 2: Comparison with state-of-the-art methods on the DOTA dataset. † means the results from AerialDetection (the same below). ‡ denotes multi-scale training and testing.

DOTA dataset, we crop the original images into 1024×1024 patches. The stride of cropping is set to 824, that is, the pixel overlap between two adjacent patches is 200. With regard to multi-scale training and testing, we first resize the original images at three scales (0.5, 1.0 and 1.5) and crop them into 1024×1024 patches with the stride of 524. We train oriented R-CNN with 12 epochs. The initial learning rate is set to 0.005 and divided by 10 at epoch 8 and 11. The ploy NMS threshold is set to 0.1 when merging image patches.

For the HRSC2016 dataset, we do not change the aspect ratios of images. The shorter sides of the images are resized to 800 while the longer sides are less than or equal to 1333. During training, 36 epochs are adopted. The initial learning rate is set to 0.005 and divided by 10 at epoch 24 and 33.

### 4.3. Evaluation of Oriented RPN

We evaluate the performance of oriented RPN in terms of recall. The results of oriented RPN are reported on the DOTA validation set, and ResNet-50-FPN is used as the backbone. To simplify the process, we just calculate the recall based on the patches cropped from original images, without merging them. The IoU threshold with ground-truth boxes is set to 0.5. We respectively select top-300, top-1000, and top-2000 proposals from each image patch to report their recall values, denoted as $\mathrm{R}_{300}$, $\mathrm{R}_{1000}$, and $\mathrm{R}_{2000}$. The results are presented in Table 1. As can be seen, our oriented RPN can achieve the recall of 92.80% when using 2000 proposals. The recall drops very slightly (0.6%) when the number of proposals changes from 2000 to 1000, but it goes down sharply when using 300 proposals. There-

| Method | Backbone | mAP(07) | mAP(12) |
|---|---|---|---|
| PIoU [4] | DLA-34 | 89.20 | - |
| DRN [28] | H-34 | - | 92.70 |
| R3Det [39] | R-101-FPN | 89.26 | 96.01 |
| DAL [27] | R-101-FPN | 89.77 | - |
| $S^2$ANet [12] | R-101-FPN | 90.17 | 95.01 |
| Rotated RPN [26] | R-101 | 79.08 | 85.64 |
| R2CNN [17] | R-101 | 73.07 | 79.73 |
| RoI Transformer [7] | R-101-FPN | 86.20 | - |
| Gliding Vertex [37] | R-101-FPN | 88.20 | - |
| CenterMap-Net [35] | R-50-FPN | - | 92.80 |
| Oriented R-CNN | R-50-FPN | 90.40 | 96.50 |
| Oriented R-CNN | R-101-FPN | **90.50** | **97.60** |

Table 3: Comparison results on the HRSC2016 dataset.

| Method | Framwork | FPS | mAP |
|---|---|---|---|
| RetinaNet-O$^\dagger$ | One-stage | **16.1** | 68.43 |
| $S^2$ANet [12] | One-stage | 15.3 | 74.12 |
| Faster R-CNN-O$^\dagger$ | Two-stage | 14.9 | 69.05 |
| RoI Transformer$^+$ | Two-stage | 11.3 | 74.61 |
| Oriented R-CNN | Two-stage | 15.1 | **75.87** |

Table 4: Speed versus accuracy on the DOTA dataset.

fore, in order to trade-off the inference speed and detection accuracy, we choose 1000 proposals as the input of oriented R-CNN head at test-time for both of the two datasets. In Figure 6, we show some examples of proposals generated by oriented RPN on the DOTA dataset. The top-200 proposals per image are displayed. As shown, our proposed oriented RPN could well localize the objects no matter their sizes, aspect ratios, directions, and denseness.

### 4.4. Comparison with State-of-the-Arts

We compare our oriented R-CNN method with 19 oriented object detection methods for the DOTA dataset, and 10 methods for the HRSC2016 dataset. Table 2 and Table 3 report the detailed comparison results on the DOTA and HRSC2016 datasets, respectively. The backbones are as follows: R-50 stands for ResNet-50, R-101 denotes ResNet-101, H-104 is the 104-layer hourglass network [38], and DLA-34 means the 34-layer deep layer aggregation network [46].

On the DOTA dataset, our method surpasses all comparison methods. With R-50-FPN and R-101-FPN as the backbones, our method obtains 75.87% and 76.28% mAP, respectively. It is surprising that using the backbone of R-50-FPN, we even outperform all comparison methods with the R-101-FPN backbone. In addition, with multi-scale training and testing strategies, our method reaches 80.87% mAP using R-50-FPN backbone, which is very competitive compared to the current state-of-the-art methods. Figure 7 shows some results on the DOTA dataset.

For the HRSC2016 dataset, we list the mAP values of

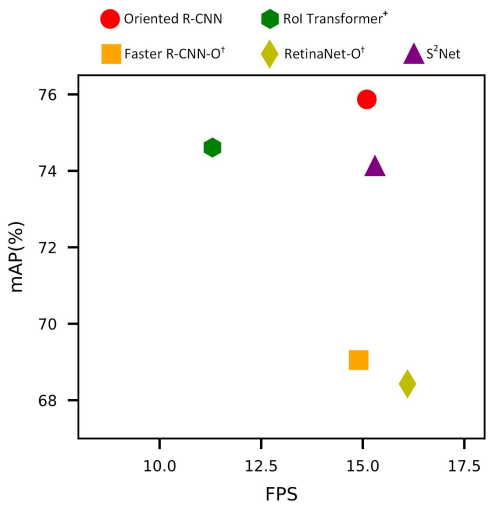


Figure 9: Speed versus accuracy on the DOTA test set.

different methods under PASCAL VOC 2007 and VOC 2012 metrics. With R-50-FPN and R-101-FPN, our oriented R-CNN both achieves the best accuracy. Some visualization results are presented in Figure 8.

### 4.5. Speed versus Accuracy

Under the same setting, we compare the speed and accuracy of different methods. The comparison results are presented in Table 4. All methods adopt R-50-FPN as the backbone. The hardware platform of testing is a single RTX 2080Ti with batch size of 1. During testing, the size of input images is $1024\times1024$. As shown in Table 4, our method has higher detection accuracy (75.87% mAP) than other methods but runs with comparable speed (15.1 FPS). The speed of oriented R-CNN is almost close to one-stage detectors, but the accuracy is much higher than one-stage detectors (see Table 4 and Figure 9).

## 5. Conclusions

This paper proposed a practical two-stage detector, named oriented R-CNN, for arbitrary-oriented object detection in images. We conduct extensive experiments on two challenging oriented object detection benchmarks. Experimental results show that our method has competitive accuracy to the current advanced two-stage detectors, while keeping comparable efficiency compared with one-stage oriented detectors.

## Acknowledgments

This work was supported by the National Key R&D Program of China under Grant 2018YFB1402600, the National Natural Science Foundation of China (No. 61772425), the Shaanxi Science Foundation for Distinguished Young Scholars (No. 2021JC-16), and the Doctorate Foundation of Northwestern Polytechnical University (No. CX2021082).

## References


[1] Seyed Majid Azimi, Eleonora Vig, Reza Bahmanyar, Marco Körner, and Peter Reinartz. Towards multi-class object detection in unconstrained remote sensing imagery. In *Proceedings of the Asian Conference on Computer Vision*, pages 150–165, 2018.

[2] Zhaowei Cai and Nuno Vasconcelos. Cascade R-CNN: Delving into high quality object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6154–6162, 2018.

[3] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

[4] Zhiming Chen, Kean Chen, Weiyao Lin, John See, Hui Yu, Yan Ke, and Cong Yang. PIoU Loss: Towards accurate oriented object detection in complex environments. In *Proceedings of the European Conference on Computer Vision*, pages 195–211, 2020.

[5] Gong Cheng, Peicheng Zhou, and Junwei Han. Learning rotation-invariant convolutional neural networks for object detection in vhr optical remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing*, 54(12):7405–7415, 2016.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[7] Jian Ding, Nan Xue, Yang Long, Gui-Song Xia, and Qikai Lu. Learning roi transformer for oriented object detection in aerial images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2849–2858, 2019.

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

[9] Kun Fu, Zhonghan Chang, Yue Zhang, and Xian Sun. Point-based estimator for arbitrary-oriented object detection in aerial images. *IEEE Transactions on Geoscience and Remote Sensing*, 59(5):4370–4387, 2021.

[10] Ross Girshick. Fast R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015.

[11] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014.

[12] J. Han, J. Ding, J. Li, and G. S. Xia. Align deep features for oriented object detection. *IEEE Transactions on Geoscience and Remote Sensing*, pages 1–11, 2021.

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2961–2969, 2017.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[15] Tong He, Zhi Tian, Weilin Huang, Chunhua Shen, Yu Qiao, and Changming Sun. An end-to-end textspotter with explicit alignment and attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5020–5029, 2018.

[16] Liping Hou, Ke Lu, Jian Xue, and Li Hao. Cascade detector with feature fusion for arbitrary-oriented objects in remote sensing images. In *Proceedings of the IEEE International Conference on Multimedia and Expo*, pages 1–6, 2020.

[17] Yingying Jiang, Xiangyu Zhu, Xiaobing Wang, Shuli Yang, Wei Li, Hua Wang, Pei Fu, and Zhenbo Luo. R2CNN: Rotational region cnn for orientation robust scene text detection. *arXiv preprint arXiv:1706.09579*, 2017.

[18] C. Li, C. Xu, Z. Cui, D. Wang, T. Zhang, and J. Yang. Feature-attentioned object detection in remote sensing imagery. In *Proceedings of the IEEE International Conference on Image Processing*, pages 3886–3890, 2019.

[19] Chengzheng Li, Chunyan Xu, Zhen Cui, Dan Wang, Tong Zhang, and Jian Yang. Feature-attentioned object detection in remote sensing imagery. In *Proceedings of the IEEE International Conference on Image Processing*, pages 3886–3890, 2019.

[20] Ke Li, Gang Wan, Gong Cheng, Liqiu Meng, and Junwei Han. Object detection in optical remote sensing images: A survey and a new benchmark. *ISPRS Journal of Photogrammetry and Remote Sensing*, 159:296–307, 2020.

[21] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2117–2125, 2017.

[22] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2980–2988, 2017.

[23] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *International Journal of Computer Vision*, 128(2):261–318, 2020.

[24] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single shot multibox detector. In *Proceedings of the European Conference on Computer Vision*, pages 21–37, 2016.

[25] Zikun Liu, Hongzhen Wang, Lubin Weng, and Yiping Yang. Ship rotated bounding box space for ship extraction from high-resolution optical satellite images with complex backgrounds. *IEEE Geoscience and Remote Sensing Letters*, 13(8):1074–1078, 2016.

[26] Jianqi Ma, Weiyuan Shao, Hao Ye, Li Wang, Hong Wang, Yingbin Zheng, and Xiangyang Xue. Arbitrary-oriented scene text detection via rotation proposals. *IEEE Transactions on Multimedia*, 20(11):3111–3122, 2018.

[27] Qi Ming, Zhiqiang Zhou, Lingjuan Miao, Hongwei Zhang, and Linhao Li. Dynamic anchor learning for arbitrary-oriented object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[28] Xingjia Pan, Yuqiang Ren, Kekai Sheng, Weiming Dong, Haolei Yuan, Xiaowei Guo, Chongyang Ma, and Changsheng Xu. Dynamic refinement network for oriented and densely packed object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11207–11216, 2020.

[29] Jiangmiao Pang, Kai Chen, Jianping Shi, Huajun Feng, Wanli Ouyang, and Dahua Lin. Libra R-CNN: Towards balanced learning for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 821–830, 2019.

[30] Wen Qian, Xue Yang, Silong Peng, Yue Guo, and Junchi Yan. Learning modulated loss for rotated object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

[31] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 779–788, 2016.

[32] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2016.

[33] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. FCOS: Fully convolutional one-stage object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9627–9636, 2019.

[34] Jinwang Wang, Jian Ding, Haowen Guo, Wensheng Cheng, Ting Pan, and Wen Yang. Mask OBB: A semantic attention-based mask oriented bounding box representation for multi-category object detection in aerial images. *Remote Sensing*, 11(24):2930, 2019.

[35] Jinwang Wang, Wen Yang, Heng-Chao Li, Haijian Zhang, and Gui-Song Xia. Learning center probability map for detecting objects in aerial images. *IEEE Transactions on Geoscience and Remote Sensing*, 59(5):4307–4323, 2021.

[36] Gui-Song Xia, Xiang Bai, Jian Ding, Zhen Zhu, Serge Belongie, Jiebo Luo, Mihai Datcu, Marcello Pelillo, and Liangpei Zhang. DOTA: A large-scale dataset for object detection in aerial images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3974–3983, 2018.

[37] Yongchao Xu, Mingtao Fu, Qimeng Wang, Yukang Wang, Kai Chen, Gui-Song Xia, and Xiang Bai. Gliding vertex on the horizontal bounding box for multi-oriented object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(4):1452–1459, 2021.

[38] Jing Yang, Qingshan Liu, and Kaihua Zhang. Stacked hourglass network for robust facial landmark localisation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 79–87, 2017.

[39] Xue Yang, Qingqing Liu, Junchi Yan, Ang Li, Zhiqiang Zhang, and Gang Yu. R3Det: Refined single-stage detector with feature refinement for rotating object. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

[40] Xue Yang and Junchi Yan. Arbitrary-oriented object detection with circular smooth label. In *Proceedings of the European Conference on Computer Vision*, pages 677–694, 2020.

[41] Xue Yang, Jirui Yang, Junchi Yan, Yue Zhang, Tengfei Zhang, Zhi Guo, Xian Sun, and Kun Fu. SCRDet: Towards more robust detection for small, cluttered and rotated objects. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8232–8241, 2019.

[42] Ze Yang, Shaohui Liu, Han Hu, Liwei Wang, and Stephen Lin. Reppoints: Point set representation for object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9657–9666, 2019.

[43] Jingru Yi, Pengxiang Wu, Bo Liu, Qiaoying Huang, Hui Qu, and Dimitris Metaxas. Oriented object detection in aerial images with box boundary-aware vectors. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, pages 2150–2159, 2021.

[44] Gongjie Zhang, Shijian Lu, and Wei Zhang. CAD-Net: A context-aware detection network for objects in remote sensing imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 57(12):10015–10024, 2019.

[45] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9759–9768, 2020.

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

[47] Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun 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.