Point2RBox: Combine Knowledge from Synthetic Visual Patterns for End-to-end Oriented Object Detection with Single Point Supervision

Yi Yu[1*], **Xue Yang**[2*], **Qingyun Li**[3,2], **Feipeng Da**[1†], **Jifeng Dai**[4,2], **Yu Qiao**[2], **Junchi Yan**[5,2†]
[1]Southeast University [2]Shanghai AI Laboratory [3]Harbin Institute of Technology
[4]Tsinghua University [5]Shanghai Jiao Tong University
{yuyi,dafp}@seu.edu.cn {yangxue,qiaoyu}@pjlab.org.cn
21b905003@stu.hit.edu.cn daijifeng@tsinghua.edu.cn yanjunchi@sjtu.edu.cn
https://github.com/yuyi1005/point2rbox-mmrotate

Abstract

With the rapidly increasing demand for oriented object detection (OOD), recent research involving weakly-supervised detectors for learning rotated box (RBox) from the horizontal box (HBox) has attracted more and more attention. In this paper, we explore a more challenging yet label-efficient setting, namely single point-supervised OOD, and present our approach called Point2RBox. Specifically, we propose to leverage two principles: 1) Synthetic pattern knowledge combination: By sampling around each labeled point on the image, we spread the object feature to synthetic visual patterns with known boxes to provide the knowledge for box regression. 2) Transform self-supervision: With a transformed input image (e.g. scaled/rotated), the output RBoxes are trained to follow the same transformation so that the network can perceive the relative size/rotation between objects. The detector is further enhanced by a few devised techniques to cope with peripheral issues, e.g. the anchor/layer assignment as the size of the object is not available in our point supervision setting. To our best knowledge, Point2RBox is the first end-to-end solution for point-supervised OOD. In particular, our method uses a lightweight paradigm, yet it achieves a competitive performance among point-supervised alternatives, 41.05%/27.62%/80.01% on DOTA/DIOR/HRSC datasets.

1. Introduction

As a fundamental task in computer vision, object detection plays an important role, e.g. in autonomous driving [8], aerial images [9, 27, 40, 43, 44], scene text [21, 25, 30, 31, 59], retail scenes [11, 32], industrial inspection [26, 39], and more, with the object detection results usually rendered in three ways: **1)** Horizontal bounding box (HBox); **2)** Rotated bounding box (RBox); **3)** Pixel-wise labels (Mask).

To teach the detector new concepts of visual objects, manual annotations are required. Early research is usually based on full supervision where the manual annotation is in the same manner as the desired network output. Although having achieved promising performance, full supervision in oriented object detection faces with two problems: **1)** RBox annotations are less available in many scenarios. In particular, a large number of datasets have already been annotated with other formats. When RBoxes are required, a possible way is to conduct labor-intensive re-annotation, e.g. DIOR-HBox [18] to DIOR-RBox [6] for aerial image (192K instances) and SKU110K-HBox [11] to SKU110K-RBox [32] for retail scene (1,733K instances). **2)** RBox annotations are much more costly. The cost of each RBox is about 36.5% higher than an HBox and 104.8% higher than a point annotation[1].

To mitigate the dependence on labor-intensive RBox labeling, H2RBox [50] and H2RBox-v2 [56] have explored the HBox-to-RBox setting that learns RBox detectors from HBox annotations. A more challenging task setting is then featured: Can we achieve oriented object detection under the weak supervision of point annotations?

Several point supervised detectors for HBox/Mask have been proposed: **1)** P2BNet [4] samples boxes of different sizes around the labeled point and classify them to achieve Point-to-HBox; **2)** Point2Mask [20] achieves segmentation using a single point annotation per target for training; **3)** SAM (Segment Anything Model) [17] produces object masks from input Point/HBox prompts. Although the above Point-to-HBox/Mask settings can be applied to Point-to-RBox, e.g. by using an additional HBox-to-RBox stage or finding the circumscribed rectangle of the mask, we show that such a solution is not perfect in both speed (7x slower

*Equal contribution. †Correspondence author. The work was partly supported by National Natural Science Foundation of China (62306069, 62222607, 72342023), Special Project on Basic Research of Frontier Leading Technology of Jiangsu Province of China (BK20192004C), China Postdoctoral Science Foundation (2023M740602), National Key R&D Program of China (2022ZD0160100). Yi Yu is also supported by Jiangsu Funding Program for Excellent Postdoctoral Talent (2023ZB616).

[1]According to https://cloud.google.com/ai-platform/data-labeling/pricing and point annotations are 1.1-1.2x more time consuming than obtaining image-level labels [1].

Figure 1. Visual detection results based on the same ResNet50 [14] backbone. The first row compares our method (Point2RBox-SK, AP_{50} = 40.27, see Table 1) with Point-to-HBox-to-RBox pipeline powered by the state-of-the-art P2BNet (2022) [4] and H2RBox-v2 (2023) [56]. The second row displays the comparison with Point-to-Mask-to-RBox method Point2Mask-RBox [20].

converting from mask [50]) and accuracy (see Fig. 1).

Motivation. Using the finger for single-point instructions is a natural way to convey the object concepts. While point supervision is gaining attention [4, 20], end-to-end point-supervised oriented detection is still a missing part of the literature. Inspired by humans' ability to use knowledge from sketches to learn real-world objects, we intend to explore Point-to-RBox with a similar and novel idea – using synthetic visual patterns for knowledge combination.

What is new? 1) A light-weight detector Point2RBox for leaning RBox from single point annotations is proposed. It has a simple and elegant structure and is trained in a single-stage end-to-end manner. **2)** At the core of Point2RBox is novel principles for RBox regression: synthetic pattern knowledge combination and transform self-supervision. The former enables the network to estimate the size and angle of real objects through the knowledge from synthetic patterns, whereas the latter to perceive the relative size/rotation between objects. **3)** As a result, Point2RBox give a competitive performance, as is displayed in Fig. 1 and Tables 1-2.

Contributions. 1) To our best knowledge, this work is the first attempt for end-to-end single point supervised OOD, where we propose two schemes: knowledge combination and self-supervision. **2)** The training pipeline and detail implementation are elucidated, with devised techniques to cope with peripheral issues, e.g. the anchor assignment when the object size is not available. **3)** Extensive experiments demonstrate the method's capability to learn RBox regression, surpassing other alternatives in performance.

2. Related Work

Beyond horizontal detection [24, 58], oriented object detection [38] has received extensive attention. Here, pproaches to oriented detection and point supervision are discussed.

Fully-supervised oriented detection. Representative works include anchor-based detector Rotated RetinaNet [23], anchor-free detector Rotated FCOS [36], and two-stage solutions, e.g. RoI Transformer [7], Oriented R-CNN [41], and ReDet [12]. Some research enhances the detector by exploiting alignment features, e.g. R^3Det [47] and S^2A-Net [13]. The angle regression may face boundary discontinuity and remedies are developed, including modulated losses [34, 45] that alleviate loss jumps, angle coders [42, 46, 55] that convert the angle into boundary-free coded data, and Gaussian-based losses [48, 49, 51, 52] transforming rotated bounding boxes into Gaussian distributions. RepPoint-based methods [15, 19, 53] provide alternatives that predict a set of sample points that bounds the spatial extent of an object.

HBox-to-RBox. The seminal work H2RBox [50] circumvents the segmentation step and achieves RBox detection directly from HBox annotation. With HBox annotations for the same object in various orientations, the geometric constraint limits the object to a few candidate angles. Supplemented with a self-supervised branch eliminating the undesired results, an HBox-to-RBox paradigm is established. An enhanced version H2RBox-v2 [56] is proposed to leverage the reflection symmetry of objects to estimate their angle, further boosting the HBox-to-RBox performance.

Some similar studies use additional annotated data for

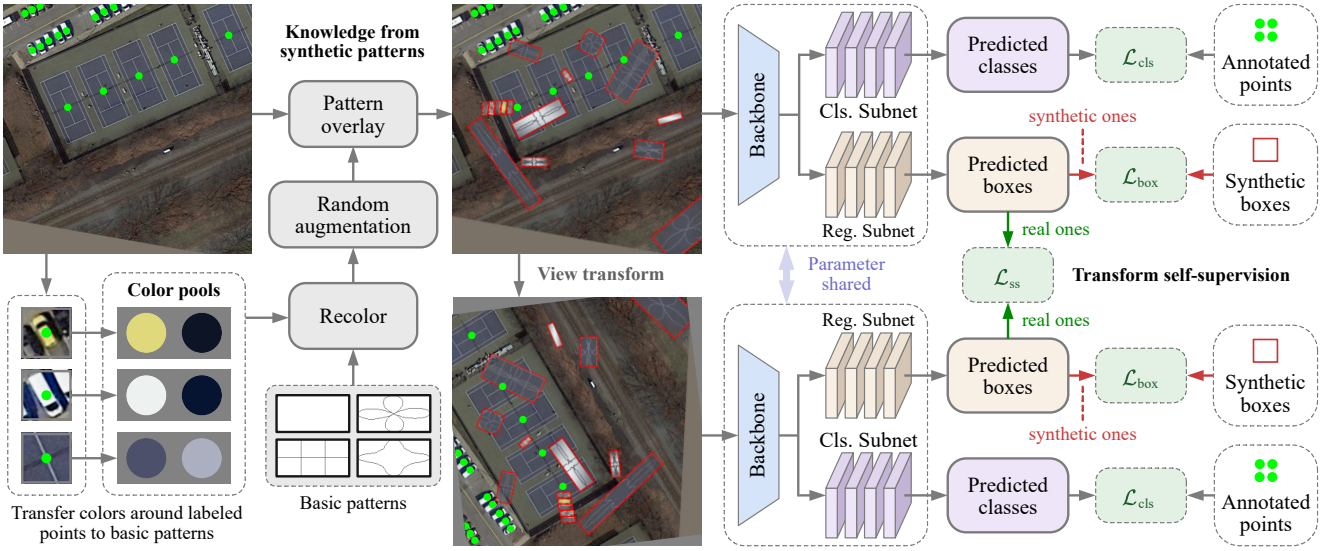

Figure 2. The training flowchart, consisting of synthetic pattern knowledge combination (Sec. 3.1) and transform self-supervision (Sec. 3.2). The core idea is to combine knowledge from synthetic patterns for size/angle estimation with that from annotated points for classification. The basic patterns are obtained based on two different settings (see Fig. 3).

training, which are also attractive but less general: **1)** OAOD [16] is proposed for weakly-supervised oriented object detection. But in fact, it uses HBox along with an object angle as annotation, which is just "slightly weaker" than RBox supervision. Such an annotation manner is not common, and OAOD is only verified on their self-collected ITU Firearm dataset. **2)** Sun et al. [35] propose a two-stage framework: i) training detector with the annotated horizontal and vertical objects, and ii) mining the rotation objects by rotating the training image to align the oriented objects as horizontally or vertically as possible. **3)** KCR [61] combines RBox-annotated source datasets with HBox-annotated target datasets, and achieves HBox-supervised oriented detection on the target datasets via transfer learning.

Point-to-HBox. Several related approaches have been developed, including: **1)** P2BNet [4] samples box proposals of different sizes around the labeled point and classify them to achieve point-supervised horizontal object detection. **2)** PSOD [10] achieves point-supervised salient object detection using an edge detector and adaptive masked flood fill.

Some methods accept partial point annotations (a common setting is 80% points and 20% HBoxes), usually termed semi-supervision: **1)** Point DETR [3] extends DETR [2] by adding a point encoder for point annotations. **2)** Group-RCNN [57] generates a group of proposals for each point annotation. **3)** CPR [54] produces center points from coarse point annotations, relaxing the supervision signals from accurate points to freely spotted points.

These Point-to-HBox methods are potentially applicable to our Point-to-RBox task setting – by using a subsequent HBox-to-RBox stage. In our experiment, the state-of-the-art methods P2BNet [4] and H2RBox-v2 [56] are used to build a Point-to-HBox-to-RBox baseline for comparison.

Point-to-Mask. Compared with point-supervised oriented detection, Point-to-Mask has been better studied. For instance, Point2Mask [20] is proposed to achieve panoptic segmentation using only a single point annotation per target for training. SAM (Segment Anything Model) [17] produces object masks from input point/HBox prompts.

The Point-to-Mask pipeline is also a potential alternative for our task – by finding the minimum circumscribed rectangle of the segmentation mask. Though the oriented bounding box can be obtained from the mask, such a complex pipeline can be less cost-efficient and perform worse [50, 56].

3. Method

An overview of the proposed Point2RBox is illustrated in Fig. 2, which consists of synthetic pattern knowledge combination (Sec. 3.1) and transform self-supervision (Sec. 3.2).

On the left is the generation procedure for the knowledge combination. In this phase, synthetic patterns are generated and randomly overlaid on the input image. These patterns with known bounding boxes are subsequently used to supervise the regression during the loss calculation.

After the image is generated, we perform a transformation on this image (randomly selected from rotate, flip, and scale). By feeding both the original view and the transformed view into the network, we obtain two sets of output RBoxes. The transform self-supervision loss is calculated between these two sets so that the output RBoxes are trained to follow the same transformation applied to the input.

In the subsequent subsections, these modules are further detailed, with the loss function described in Sec. 3.3.

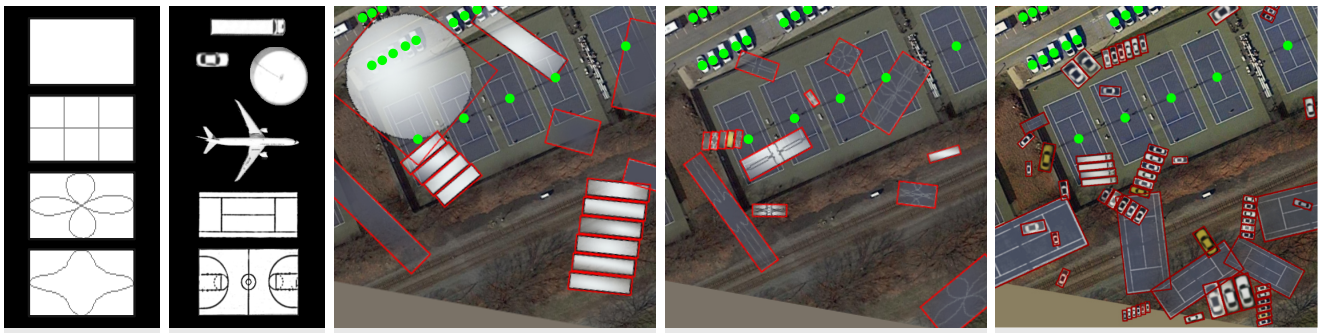

Figure 3. Two settings of obtaining basic patterns (see Sec. 3.1) and the illustration of training images overlaid with synthetic patterns. SetRC: Rectangles and circles with curve textures. SetSK: One simple sketch pattern for each category (see Table 3 for ablation).

3.1. Synthetic Pattern Knowledge Combination

In single point-supervised OOD, we do not know the exact size/angle of the labeled object. Yet we can generate a new one – with a similar texture and known bounding box. Based on this idea, we sample around each labeled point, extract its neighbor color feature, spread them to basic patterns, and overlay them on the input image. These patterns are then used to enable the training of the regression subnet.

Basic patterns. 1) SetRC *(point supervision)*. It consists of rectangles and circles with white filling and black edges, sized 160×160, 160×80, 160×40, 80×80, 80×40, and 80×20. The above preset is used to keep the size of generated patterns in a reasonable range. **2) SetSK** *(point supervision with one-shot samples)*. It consists of one sketch pattern for each category (e.g. only fifteen for DOTA-v1.0). The patterns are manually cropped from the first several training images and are adjusted to "white surfaces and black edges". It takes just about ten minutes to obtain all fifteen patterns for the DOTA-v1.0 dataset.

In the meanwhile, curve patterns can be overlaid on the shapes to improve the diversity (in SetRC, see Fig. 3). Two types of textures are used: **1)** One to four equally spaced lines parallel to bounding boxes. **2)** The curve defined by the following polar coordinate equation:

$$\rho = (1-k)\left|\cos^{n}(2\theta)\right| + k \tag{1}$$

where n is a random number in $[0, 8]$, and k is a random number in $[0.1, 0.6]$, both from uniform distribution. The two types of textures are adopted each with a probability of 0.5. Several example patterns are displayed in Fig. 3.

Color sampling. The face color C_{face} and the edge color C_{edge} around each labeled point are extracted as follows:

$$\begin{cases} C_{\text{face}} = \text{mean}\left(I_0\right) \\ C_{\text{edge}} = \text{sum}\left(dI_1\right) \end{cases} \tag{2}$$

where I_0 and I_1 are the neighbors around a labeled point. We simply use 5×5 for I_0 and 33×33 for I_1. Here d is the gradient of I_1 indicating the edge intensity of each pixel (the sum of d is uniformed to 1).

Recolor. We use a gray-scale image as the basic pattern, which can be denoted as P, with its value in the range $(0, 1)$. The recolor step can be expressed as:

$$P_{\text{recolor}} = PC_{\text{face}} + (1-P)\,C_{\text{edge}} \tag{3}$$

This formula maps the extracted color C_{face} and C_{edge} to the basic pattern. By this means, the diversity of the patterns is significantly enriched, so that the trained regression subnet can better estimate the RBox of the real data.

Random augmentation. The recolored patterns are first augmented with the random flip, resize, and rotation. Afterward, they are moved to a random position inside the image border. To avoid overlapping patterns, NMS (Non-Maximum Suppression) is then applied so that the IoU (Intersection over Union) between patterns is less than 0.05.

The random resize can be formulated as:

$$\begin{cases} s_{\text{base}} = \exp\left(\sigma_{\text{base}}\text{randn}()\right) \\ w = s_{\text{base}} \exp\left(\sigma_w \text{randn}()\right) w_0 \\ h = \frac{h_0}{w_0} \exp\left(\sigma_r \text{randn}()\right) w \end{cases} \tag{4}$$

where randn() generates random numbers from the standard normal distribution; w_0 and h_0 are the original pattern sizes; w and h are the resized ones; s_{base} is a base scale for all patterns in the same image; σ_{base}, σ_w, and σ_r control the variation of scale, width, and ratio, set to 0.4 by default.

For the other augmentation i.e. random flip and rotation, the probability is set to 0.5 and 1, respectively.

In particular, to avoid the real objects being completely occluded, we use transparent blending:

$$t(x, y) = \exp\left(-ax^2 - by^2\right) \times t_1 + t_0 \tag{5}$$

where $t(x, y)$ is the opacity channel of the synthetic pattern; x and y are coordinates in range $[-1, 1]$; a and b are random numbers in $[0.1, 2]$ from uniform distribution; $t_0 = 0.1$ and $t_1 = 0.9$ keep the opacity between 10% to 100%.

In addition, some of the patterns will be randomly selected to produce a set of tightly arranged patterns through

translation, which can also be observed in Fig. 3.

Finally, these generated patterns are overlaid on the original image and their known bounding boxes are used for training, providing the knowledge for box regression.

3.2. Transform Self-supervision

Our training flowchart has two parameter-shared branches (see Fig. 2), namely the original branch and the transformed branch (i.e. with a transformed input). The self-supervision is performed between the two branches to supervise each other. The input of the transformed branch can be set as:

$$I_{\text{trs}} = transform\left(I_{\text{ori}}\right) \tag{6}$$

where I_{ori} is the input of the original branch; I_{trs} is the input of the transformed branch; $transform\left(\cdot\right)$ is a transformation randomly selected from flip, rotate, and scale. The probability of scale is set to 30%, and the proportion between rotate and flip is set to 95:5 adopted from [56].

The output RBoxes of the two branches are defined as:

$$\begin{cases} B_{\text{ori}} = (x_{\text{ori}}, y_{\text{ori}}, w_{\text{ori}}, h_{\text{ori}}, \theta_{\text{ori}}) \\ B_{\text{trs}} = (x_{\text{trs}}, y_{\text{trs}}, w_{\text{trs}}, h_{\text{trs}}, \theta_{\text{trs}}) \end{cases} \tag{7}$$

where B_{ori} is the output RBoxes of original branch; B_{trs} is those of transformed branch.

The two branches supervise each other by minimizing the loss between B_{ori} and B_{trs} so that the transformation between B_{ori} and B_{trs} is trained to follow the same transformation being applied to the input image.

It should be noted that only the output RBoxes assigned to real objects (i.e. point-annotated ones) are involved in transform self-supervision, whereas synthetic ones are directly supervised by their known bounding boxes.

Flip. When the input image is vertically flipped, the angle of output RBoxes is also supposed to be flipped. Therefore, the self-supervised loss can be expressed as:

$$L_{\text{flp}}\left(B_{\text{ori}}, B_{\text{trs}}\right) = smooth_{L1}\left(mod(\theta_{\text{trs}} + \theta_{\text{ori}}), 0\right) \tag{8}$$

where θ_{ori} is the output angle of original branch; θ_{trs} is the output angle of transformed branch; $mod(x) = (x + \pi/2 \mod \pi) - \pi/2$ limits the difference in range $[-\pi/2, \pi/2)$.

Rotate. When the input image is rotated by $\mathcal{R}$, the angle of output RBoxes is supposed to be likewise rotated. Therefore, the self-supervised loss can be expressed as:

$$L_{\text{rot}}\left(B_{\text{ori}}, B_{\text{trs}}\right) = smooth_{L1}\left(mod(\theta_{\text{trs}} - \theta_{\text{ori}}), \mathcal{R}\right) \tag{9}$$

where θ_{ori}, θ_{trs}, and $mod(x)$ share the same definition as Eq. (8); $\mathcal{R}$ is the rotation angle being applied to the input image in range $(0.25\pi, 0.75\pi)$ adopted from [56].

Scale. When the input image is scaled by s, the center and the size of output RBoxes should be likewise scaled. Therefore, the self-supervised loss can be expressed as:

$$L_{\text{sca}}\left(B_{\text{ori}}, B_{\text{trs}}\right) = GIoU\left(r2h(B_{\text{ori}}) \times s, r2h(B_{\text{trs}})\right) \tag{10}$$

where B_{ori} and B_{trs} are outputs of the original and transformed branches; $r2h\left(\cdot\right)$ is the function to get the circumscribed HBoxes of RBoxes, s is the scaling factor being applied to the input image in range $(0.5, 1.5)$.

3.3. Loss Functions

The predicted RBoxes and the corresponding ground-truths are denoted as B_{pred} and B_{gt}.

Loss from point supervision. Point annotations are used to train the classification and the center of RBoxes:

$$\begin{cases} \mathcal{L}_{\text{cls}} = L_{\text{cls}}\left(M_{\text{point}} c_{\text{pred}}, M_{\text{point}} c_{\text{gt}}\right) \\ \mathcal{L}_{\text{cen}} = L_1\left(M_{\text{point}} xy_{\text{pred}}, M_{\text{point}} xy_{\text{gt}}\right) \end{cases} \tag{11}$$

where M_{point} is a mask to select those RBoxes that are assigned to annotated points; c_{pred} and xy_{pred} are the predicted classification scores and centers; c_{gt} and xy_{gt} are the labels and coordinates of annotated points; L_{cls} is the classification loss defined by backbone oriented detector.

Loss from knowledge combination. Knowledge from synthetic patterns is combined to learn the box regression:

$$\mathcal{L}_{\text{box}} = L_{\text{box}}\left(M_{\text{box}} B_{\text{pred}}, M_{\text{box}} B_{\text{gt}}\right) \tag{12}$$

where M_{box} is a mask to select those RBoxes that are assigned to synthetic boxes; L_{box} is the loss function depending on the backbone oriented detector.

Loss from transform self-supervision. The loss is calculated between the original and transformed branches and the loss function is selected corresponding to the type of transformation being applied to the input image:

$$\mathcal{L}_{\text{ss}} = L_{\text{flp/rot/sca}}\left(M_{\text{ori}} M_{\text{point}} B_{\text{pred}}, M_{\text{trs}} M_{\text{point}} B_{\text{pred}}\right) \tag{13}$$

where $L_{\text{flp/rot/sca}}$ is defined by Eqs. (8, 9, 10); M_{ori} and M_{trs} are the masks to select those RBoxes from the original branch or transformed branch.

Overall loss. The overall loss of the proposed network is the weighted sum of the above losses:

$$\mathcal{L}_{\text{total}} = \omega_{\text{cls}}\mathcal{L}_{\text{cls}} + \omega_{\text{cen}}\mathcal{L}_{\text{cen}} + \omega_{\text{box}}\mathcal{L}_{\text{box}} + \omega_{\text{ss}}\mathcal{L}_{\text{ss}} \tag{14}$$

where ω_{cls}, ω_{cen}, ω_{box} are set to 1, 0.1, 1 by default, and ω_{ss} is set to 0.3 (flip/rotate) or 0.02 (scale).

3.4. Label Assignment

Currently, available detectors largely rely on FPN (Feature Pyramid Network) [22]. For example, Rotated FCOS [36] usually uses five feature layers, with large and small objects assigned to different ones. While point annotations do not provide any size information, they do not apply to such an FPN-based assignment strategy.

In terms of YOLOF [5] that barely uses a one-level feature, it uses five preset anchors with sizes 32, 64, 128, 256, and 512. Therefore its anchor assignment strategy is also incompatible with point annotations.

To mitigate this problem, we propose a classification-score-based assignment rule. Instead of assigning ground-

truths to the anchor with the highest IoU, we assign them (including both labeled points and synthetic boxes) to the one that produces the highest classification score.

Specifically, YOLOF is used as the backbone detector. The anchor stride is 16 and all five anchors are set to a fixed size (64×64 for the DOTA dataset and 128×128 for others). Then the matching scores between anchors and ground-truths can be calculated as:

$$score = \begin{cases} 0, & L_1\left(xy_{\text{pred}}, xy_{\text{gt}}\right) > 32 \\ L_{\text{cls}}\left(c_{\text{pred}}, c_{\text{gt}}\right), & otherwise \end{cases} \tag{15}$$

where xy_{pred} and xy_{gt} are the center coordinates of predicted boxes and ground-truths; c_{pred} and c_{gt} are the predicted classification scores and ground-truth labels.

Afterward, following the setting of YOLOF, we use K-nearest to find four positive anchors with the highest scores for each ground truth. The improvement of the above design is verified in the ablation study (see Sec. 4.3).

3.5. Inference Phase

Although using a synthetic pattern generation module and two branches in training (see Fig. 2), Point2RBox does not require these operations in inference. Due to the parameter sharing of the two branches, the inference only involves the forward propagation of the backbone, the classification head, and the regression head. Thus, it has a similar inference speed compared to the backbone detector that it is based on.

4. Experiments

Experiments are carried out on NVIDIA RTX4090/A100 GPUs using PyTorch 1.13.1 [33] and the rotation detection tool kits: MMRotate 1.0.0 [60]. All the experiments follow the same hyper-parameters (learning rate, batch size, optimizer, etc.).

4.1. Settings and Datasets

We use the FPN-free detector YOLOF [5] with a fixed anchor size (64 for DOTA and 128 for DIOR/HRSC) as the baseline method for developing our Point2RBox. Such a choice is mainly upon the fact that positive samples (annotated with points) cannot be assigned to different FPN layers or different anchors based on their annotated sizes.

Average precision (AP) is adopted as the primary metric to compare our methods with existing alternatives. Unless otherwise specified, all the listed models are configured based on ResNet50 [14] backbone. All models are trained with AdamW [28], with an initial learning rate of 5e-5 and a mini-batch size of 4. Besides, we adopt a learning rate warm-up for 500 iterations, and the learning rate is divided by ten at each decay step. "1x" and "6x" schedules indicate 12 and 72 epochs for training, and "RR" denotes random rotation augmentation. "1x" is used for the DOTA and DIOR datasets and "6x+RR" for HRSC. Random flipping and shifting are always adopted by default. For a fair comparison, all results are evaluated without multi-scale technique [60].

DOTA [40]. DOTA-v1.0 contains 2,806 aerial images—1,411 for training, 937 for validation, and 458 for testing, as annotated using 15 categories with 188,282 instances in total. The categories are defined as: Plane (PL), Baseball Diamond (BD), Bridge (BR), Ground Track Field (GTF), Small Vehicle (SV), Large Vehicle (LV), Ship (SH), Tennis Court (TC), Basketball Court (BC), Storage Tank (ST), Soccer-Ball Field (SBF), Roundabout (RA), Harbor (HA), Swimming Pool (SP), and Helicopter (HC). We follow the default preprocessing in MMRotate: The high-resolution images are split into 1,024 $\times$ 1,024 patches with an overlap of 200 pixels for training, and the detection results of all patches are merged to evaluate the performance.

DIOR [6]. DIOR-RBox is an aerial image dataset re-annotated with RBoxes based on its original HBox-annotated version DIOR [18]. There are 23,463 images and 190,288 instances with 20 classes. DIOR-RBox has a high variation in object size with high intra-class diversity.

HRSC [27]. It contains ship instances both on the sea and inshore, with arbitrary orientations. The training, validation, and testing set includes 436, 181, and 444 images, respectively. With preprocessing by MMRotate, images are scaled to 800 $\times$ 800 for training/testing.

4.2. Main Results

DOTA-v1.0. Table 1 shows that Point2RBox outperforms currently available two-stage solution – Point-to-HBox-to-RBox, even if the Point-to-HBox-to-RBox is powered by the state-of-the-art methods P2BNet [4] (Point-to-HBox) and H2RBox-v2 [56] (HBox-to-RBox).

Since point annotations cannot be assigned to different FPN layers or different anchors by size, our Point2RBox is based on the FPN-free method YOLOF-A1 [5]. The performance gap between Point2RBox and the RBox-supervised YOLOF-A1 baseline is 9.67% (40.27% vs. 49.94%), proving the effectiveness of combining knowledge from synthetic patterns for point-supervised oriented object detection.

Barely using synthetic patterns (i.e. SetRC), Point2RBox achieves AP_{50} of 34.07%. While one sketch pattern for each category is cooperated (only fifteen patterns in total, i.e. SetSK), the performance is further boosted to 40.27%. By utilizing a stronger backbone CSPNeXt-l [29], Point2RBox obtains 41.05% on DOTA-v1.0.

In the last row of Table 1, we additionally explore a two-stage training pipeline. In the first stage, Point2RBox-SK (based on YOLOF-A1) is trained to generate pseudo RBoxes labels. These pseudo labels enable the anchor-based assignment in the second stage (based on YOLOF-A5), which improves the performance to 44.90%.

DIOR. Table 2 shows that Point2RBox achieves AP_{50} of 24.66% and 27.34% in RC and SK settings respectively.

Methods	PL	BD	BR	GTF	SV	LV	SH	TC	BC	ST	SBF	RA	HA	SP	HC	AP_{50}
RBox-supervised																
RepPoints (2019) [53]	86.7	81.1	41.6	62.0	76.2	56.3	75.7	90.7	80.8	85.3	63.3	66.6	59.1	67.6	33.7	68.45
RetinaNet (2017) [23]	88.2	77.0	45.0	69.4	71.5	59.0	74.5	90.8	84.9	79.3	57.3	64.7	62.7	66.5	39.6	68.69
GWD (2021) [48]	89.3	75.4	47.8	61.9	79.5	73.8	86.1	90.9	84.5	79.4	55.9	59.7	63.2	71.0	45.4	71.66
FCOS (2019) [36]	89.1	76.9	50.1	63.2	79.8	79.8	87.1	90.4	80.8	84.6	59.7	66.3	65.8	71.3	41.7	72.44
S^2A-Net (2022) [13]	89.2	83.0	52.5	74.6	78.8	79.2	87.5	90.9	84.9	84.8	61.9	68.0	70.7	71.4	59.8	**75.81**
YOLOF-A5 (2021) [5] [1]	86.9	71.7	44.9	62.7	69.9	62.0	74.1	90.9	78.4	72.9	47.1	60.7	61.3	65.3	50.2	66.54
YOLOF-A1 (2021) [5] [1]	68.7	52.3	25.2	36.8	50.6	47.8	58.7	81.6	64.5	65.4	16.9	56.0	30.0	51.4	43.5	49.94
HBox-to-RBox																
Sun et al. (2021) [35]	51.5	38.7	16.1	36.8	29.8	19.2	23.4	83.9	50.6	80.0	18.9	50.2	25.6	28.7	25.5	38.60
BoxInst-RBox (2021) [37] [2]	68.4	40.8	33.1	32.3	46.9	55.4	56.6	79.5	66.8	82.1	41.2	52.8	52.8	65.0	30.0	53.59
H2RBox (2023) [50]	88.5	73.5	40.8	56.9	77.5	65.4	77.9	90.9	83.2	85.3	55.3	62.9	52.4	63.6	43.3	67.82
H2RBox-v2 (2023) [56]	89.0	74.4	50.0	60.5	79.8	75.3	86.9	90.9	85.1	85.0	59.2	63.2	65.2	70.5	49.7	**72.31**
Point-to-RBox																
Point2Mask-RBox (2023) [20] [2]	4.0	23.1	3.8	1.3	15.1	1.0	3.3	19.0	1.0	29.1	0.0	9.5	7.4	21.1	7.1	9.72
P2BNet+H2RBox (2023) [4, 50] [3]	24.7	35.9	7.1	27.9	3.3	12.1	17.5	17.5	0.8	34.0	6.3	49.6	11.6	27.2	18.8	19.63
P2BNet+H2RBox-v2 (2023) [4, 56] [3]	11.0	44.8	14.9	15.4	36.8	16.7	27.8	12.1	1.8	31.2	3.4	50.6	12.6	36.7	12.5	21.87
Point2RBox-RC (ours) [4]	62.9	64.3	14.4	35.0	28.2	38.9	33.3	25.2	2.2	44.5	3.4	48.1	25.9	45.0	22.6	34.07
Point2RBox-SK (ours) [4]	53.3	63.9	3.7	50.9	40.0	39.2	45.7	76.7	10.5	56.1	5.4	49.5	24.2	51.2	33.8	40.27
Point2RBox-SK (CSPNeXt) [4]	63.9	49.9	11.7	48.4	42.2	43.4	51.5	90.6	3.0	53.3	3.0	45.1	20.5	50.9	38.2	41.05
Point2RBox-SK (two-stage) [4,5]	66.4	59.5	5.2	52.6	54.1	53.9	57.3	90.8	3.2	57.8	6.1	47.4	22.9	55.7	40.5	**44.90**

[1] A5: Using five anchor sizes (16, 32, 64, 128, 256). A1: Using single anchor size (64). Our Point2RBox is based on YOLOF-A1.
[2] -RBox: The minimum rectangle operation is performed on the output Mask to obtain the RBox.
[3] Using P2BNet (2022) [4] for Point-to-HBox and then H2RBox-v2 (2023) [56] for HBox-to-RBox.
[4] RC: Using rectangles and circles with curve textures as basic patterns. SK: Using one sketch pattern for each category as basic patterns.
[5] two-stage: Point2RBox-SK (A1) is trained to generate pseudo RBoxes, which enable the anchor-based assignment in the second stage (A5).

Table 1. Detection performance of each category on the DOTA-v1.0 and the mean AP_{50} of all categories.

Methods	DIOR	HRSC
RBox-supervised		
RetinaNet (2017) [23]	54.60	84.49
GWD (2021) [48]	57.80	86.67
FCOS (2019) [36]	**58.60**	88.99
YOLOF-A5 (2021) [5] [1]	48.01	**89.44**
YOLOF-A1 (2021) [5] [2]	37.51	81.14
HBox-to-RBox		
H2RBox (2023) [50]	**57.00**	7.03
KCR (2023) [61]	-	79.10
H2RBox-v2 (2023) [56]	56.92	**89.66**
Point-to-RBox		
Point2Mask-RBox (2023) [20]	13.77	29.95
P2BNet+H2RBox (2023) [4, 50]	22.59	-
P2BNet+H2RBox-v2 (2023) [4, 56]	23.61	14.60
Point2RBox-RC (ours)	24.66	78.77
Point2RBox-SK (ours)	27.34	79.40
Point2RBox-SK (CSPNeXt)	**27.62**	**80.01**

[1] A5: Using five anchor sizes (16, 32, 64, 128, 256).
[2] A1: Using single anchor size (128).

Table 2. AP_{50} on the DIOR and the HRSC datasets.

Compared with the state-of-the-art Point-to-HBox-to-RBox solution (i.e. P2BNet [4] + H2RBox-v2 [56]), our method uses a light-weight end-to-end paradigm, yet obtains a competitive performance (27.34% vs. 22.30%).

HRSC. Table 2 shows that Point2RBox achieves 78.77% in the SetRC and 79.40% in the SetSK. Notably, HRSC only has one category, so we also use only one pattern cropped from the first training image as the basic pattern for the SK setting. A previous work KCR (2023) [61] combines knowledge from RBox-annotated DOTA dataset to achieve HBox-to-RBox on HRSC. Compared with that, our SetSK method, under a more challenging Point-to-RBox setting, outperforms KCR by 0.3% (79.40% vs. 79.10%).

Compared to RBox-supervised counterpart YOLOF-A1 [5], the performance gap on HRSC is only 1.74% (79.40% vs. 81.14%). Finally, Point2RBox obtains 80.01% on HRSC by further utilizing a stronger CSPNeXt-l [29] backbone.

Computational cost. Point2RBox is light-weight in three folds: **1)** end-to-end; **2)** fewer parameters; **3)** anchor/FPN-free. Specifically, training Point2RBox-SK on DOTA takes about 5 hours and the inference speed is about 112 fps on our Intel i9-14900 + NVIDIA RTX4090 hardware.

4.3. Ablation Studies

Several ablation studies are performed on Point2RBox to evaluate the impact of each proposed module.

Basic pattern settings. Table 4 studies the impact of different strategies to obtain basic patterns for synthetic generation. The "Shape" column indicates using rectangles and

Datasets	Shape	Curve	Sketch	AP_{50}
DOTA [40]	✓			29.72
	✓	✓		34.07
			✓	**40.27**
DIOR [6]	✓			22.23
	✓	✓		24.66
			✓	**27.34**
HRSC [27]	✓			75.01
	✓	✓		78.77
			✓	**79.40**

Table 3. Ablation with different basic pattern settings.

circles with white filling and black edges; the "Curve" column indicates adding curve textures on the rectangles and circles; the "Sketch" column indicates using one sketch pattern for each category (see Sec. 3.1). The three rows in Table 4 correspond to the three cases displayed in Fig. 3

Even if using simplest rectangles and circles, we show the knowledge can be well combined, resulting in AP_{50} of 75.01% on HRSC. When curve textures are added (i.e. SetRC), the performance is further boosted to 78.77%, proving the effectiveness of using curve textures. Sketch patterns (i.e. SetSK) provide more accurate semantic boundaries, leading to an improvement of 6.20%/2.68%/0.63% on DOTA/DIOR/HRSC compared to the SetRC.

Transform self-supervision. Table 4 studies the impact of using different transformations in the transform self-supervision (see Sec. 3.2). Rotation and flipping are adopted together since they both act on the angle. According to the results, when self-supervised by rotated and flipped views, the performance is boosted by 4.29% on average, whereas incorporating scaling gains an additional 1.82% improvement. The results prove that the transform self-supervision plays a crucial role in the proposed training paradigm.

Annotation inaccuracy. Table 5 offsets the coordinates of annotated points by a noise from the uniform distribution $[-\sigma H, +\sigma H]$, where H is the height of objects. Such a setting is designed to simulate the noise in the real annotation. When $\sigma = 10\%$, AP_{50} on the DIOR dataset is even improved to 30.82%. For some categories, e.g. basketball court with a circle in the center, the network sometimes takes the size of the central circle as the size of the basketball court. Adding noise alleviates this issue, which may explain the performance improvement. When $\sigma = 20\%$, the AP_{50} of Point2RBox drops by only 1.03% on average, which demonstrates the robustness of our method.

Label assignment. With a fixed anchor size, AP_{50} of using one/three/five anchors is 37.15%/39.48%/40.27% on DOTA, which proves that using multiple anchors of the same size (assigned based on classification score, see Sec. 3.4) can improve performance to some extent.

Knowledge combination. Our method's effectiveness lies in spreading features near each labeled point to the generated patterns, which narrows the gap between the synthetic and real data. With this key recolor step removed (i.e. directly pasting augmented patterns like copy-paste), the AP_{50} is much lower (40.27% vs. 28.72%) on DOTA (SetSK).

Datasets	Rotate	Flip	Scale	AP_{50}
DIOR [6]				21.34
	✓	✓		24.97
	✓	✓	✓	**27.34**
HRSC [27]				73.18
	✓	✓		78.13
	✓	✓	✓	**79.40**

Table 4. Ablation with the transform self-supervision.

Datasets	σ = 0%	σ = 10%	σ = 20%
DOTA [40]	**40.27**	39.60	38.42
DIOR [6]	27.34	**30.82**	27.22
HRSC [27]	**79.40**	78.81	78.28

Table 5. Ablation with the inaccuracy in point annotations.

5. Conclusion

This paper has presented Point2RBox, a weakly-supervised oriented object detector that learns from the point annotation. It adopts an end-to-end paradigm to directly obtain the RBox prediction through knowledge combination from synthetic visual patterns, which has the advantage of being concise and cost-efficient over the two-stage alternatives (e.g. Point-to-HBox-to-RBox or generating RBox pseudo labels from the Mask). Supplemented with the transform self-supervision, the performance is further improved.

Experiments are carried out with the following observations: **1)** Point2RBox achieves point-supervised oriented object detection in an end-to-end training manner. Upon that, we show the knowledge from synthetic patterns can be combined to estimate the size and angle of real objects. **2)** Compared with KCR [61] that combines knowledge for HBox-supervised setting, our method outperforms KCR by 0.3% (HRSC: 79.40% vs. 79.10%) under a more challenging point-supervised setting. **3)** Our method outperforms the state-of-the-art alternative (i.e. P2BNet [4] + H2RBox-v2 [56]) by a large margin based on the same ResNet50 backbone (DOTA/DIOR/HRSC: 40.27%/27.34%/79.40% vs. 21.87%/22.30%/14.60%). With CSPNeXt-l [29] backbone, the performance reaches 41.05%/27.62%/80.01%, proving the super effectiveness of the proposed method.

Limitations. 1) Point2RBox gives a bad performance on some categories (i.e. BR/BC/SBF), mainly due to their non-unique boundaries. For instance, the central circle of the basketball court is often mistaken as the boundary. **2)** Point2RBox can only be built upon FPN-free detectors (e.g. YOLOF-A1) since points cannot be assigned to different FPN layers/anchors based on their sizes.

References

[1] Amy Bearman, Olga Russakovsky, Vittorio Ferrari, and Li Fei-Fei. What's the point: Semantic segmentation with point supervision. In *European Conference on Computer Vision*, pages 549–565, 2016. 1

[2] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, 2020. 3

[3] Liangyu Chen, Tong Yang, Xiangyu Zhang, Wei Zhang, and Jian Sun. Points as queries: Weakly semi-supervised object detection by points. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8819–8828, 2021. 3

[4] Pengfei Chen, Xuehui Yu, Xumeng Han, Najmul Hassan, Kai Wang, Jiachen Li, Jian Zhao, Humphrey Shi, Zhenjun Han, and Qixiang Ye. Point-to-box network for accurate object detection via single point supervision. In *European Conference on Computer Vision*, 2022. 1, 2, 3, 6, 7, 8

[5] Qiang Chen, Yingming Wang, Tong Yang, Xiangyu Zhang, Jian Cheng, and Jian Sun. You only look one-level feature. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13034–13043, 2021. 5, 6, 7

[6] Gong Cheng, Jiabao Wang, Ke Li, Xingxing Xie, Chunbo Lang, Yanqing Yao, and Junwei Han. Anchor-free oriented proposal generator for object detection. *IEEE Transactions on Geoscience and Remote Sensing*, 2022. 1, 6, 8

[7] Jian Ding, Nan Xue, Yang Long, Gui-Song Xia, and Qikai Lu. Learning roi transformer for oriented object detection in aerial images. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2849–2858, 2019. 2

[8] Di Feng, Christian Haase-Schütz, Lars Rosenbaum, Heinz Hertlein, Claudius Gläser, Fabian Timm, Werner Wiesbeck, and Klaus Dietmayer. Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. *IEEE Transactions on Intelligent Transportation Systems*, 22(3):1341–1360, 2021. 1

[9] Kun Fu, Zhonghan Chang, Yue Zhang, Guangluan Xu, Keshu Zhang, and Xian Sun. Rotation-aware and multi-scale convolutional neural network for object detection in remote sensing images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 161:294–308, 2020. 1

[10] Shuyong Gao, Wei Zhang, Yan Wang, Qianyu Guo, Chenglong Zhang, Yangji He, and Wenqiang Zhang. Weakly-supervised salient object detection using point supervision. In *AAAI Conference on Artificial Intelligence*, pages 670–678, 2022. 3

[11] Eran Goldman, Roei Herzig, Aviv Eisenschtat, Jacob Goldberger, and Tal Hassner. Precise detection in densely packed scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5227–5236, 2019. 1

[12] Jiaming Han, Jian Ding, Nan Xue, and Gui-Song Xia. Redet: A rotation-equivariant detector for aerial object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2785–2794, 2021. 2

[13] Jiaming Han, Jian Ding, Jie Li, and Gui-Song Xia. Align deep features for oriented object detection. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–11, 2022. 2, 7

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 2, 6

[15] Liping Hou, Ke Lu, Xue Yang, Yuqiu Li, and Jian Xue. G-rep: Gaussian representation for arbitrary-oriented object detection. *arXiv preprint arXiv:2205.11796*, 2022. 2

[16] Javed Iqbal, Muhammad Akhtar Munir, Arif Mahmood, Afsheen Rafaqat Ali, and Mohsen Ali. Leveraging orientation for weakly supervised object detection with application to firearm localization. *Neurocomputing*, 440:310–320, 2021. 3

[17] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollar, and Ross Girshick. Segment anything. In *IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023. 1, 3

[18] 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. 1, 6

[19] Wentong Li, Yijie Chen, Kaixuan Hu, and Jianke Zhu. Oriented reppoints for aerial object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1829–1838, 2022. 2

[20] Wentong Li, Yuqian Yuan, Song Wang, Jianke Zhu, Jianshu Li, Jian Liu, and Lei Zhang. Point2mask: Point-supervised panoptic segmentation via optimal transport. In *IEEE/CVF International Conference on Computer Vision*, 2023. 1, 2, 3, 7

[21] Minghui Liao, Zhen Zhu, Baoguang Shi, Gui-Song Xia, and Xiang Bai. Rotation-sensitive regression for oriented scene text detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5909–5918, 2018. 1

[22] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 936–944, 2017. 5

[23] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2):318–327, 2020. 2, 7

[24] 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. 2

[25] Xuebo Liu, Ding Liang, Shi Yan, Dagui Chen, Yu Qiao, and Junjie Yan. Fots: Fast oriented text spotting with a unified network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5676–5685, 2018. 1

[26] Yuekai Liu, Hongli Gao, Liang Guo, Aoping Qin, Canyu Cai, and Zhichao You. A data-flow oriented deep ensemble learning method for real-time surface defect inspection. *IEEE Transactions on Instrumentation and Measurement*, 69(7): 4681–4691, 2020. 1

[27] Zikun Liu, Liu Yuan, Lubin Weng, and Yiping Yang. A high resolution optical satellite image dataset for ship recognition

and some new baselines. In *International Conference on Pattern Recognition Applications and Methods*, pages 324–331, 2017. 1, 6, 8

[28] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018. 6

[29] Chengqi Lyu, Wenwei Zhang, Haian Huang, Yue Zhou, Yudong Wang, Yanyi Liu, Shilong Zhang, and Kai Chen. Rtmdet: An empirical study of designing real-time object detectors. *arXiv preprint arXiv:2212.07784*, 2022. 6, 7, 8

[30] 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. 1

[31] Nibal Nayef, Fei Yin, Imen Bizid, Hyunsoo Choi, Yuan Feng, Dimosthenis Karatzas, Zhenbo Luo, Umapada Pal, Christophe Rigaud, Joseph Chazalon, Wafa Khlif, Muhammad Muzzamil Luqman, Jean-Christophe Burie, Cheng-lin Liu, and Jean-Marc Ogier. Icdar2017 robust reading challenge on multi-lingual scene text detection and script identification - rrc-mlt. In *IAPR International Conference on Document Analysis and Recognition*, pages 1454–1459, 2017. 1

[32] 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11207–11216, 2020. 1

[33] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035, 2019. 6

[34] Wen Qian, Xue Yang, Silong Peng, Junchi Yan, and Yue Guo. Learning modulated loss for rotated object detection. In *AAAI Conference on Artificial Intelligence*, pages 2458–2466, 2021. 2

[35] Yongqing Sun, Jie Ran, Feng Yang, Chenqiang Gao, Takayuki Kurozumi, Hideaki Kimata, and Ziqi Ye. Oriented object detection for remote sensing images based on weakly supervised learning. In *IEEE International Conference on Multimedia & Expo Workshops*, pages 1–6. IEEE, 2021. 3, 7

[36] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *IEEE/CVF International Conference on Computer Vision*, pages 9626–9635, 2019. 2, 5, 7

[37] Zhi Tian, Chunhua Shen, Xinlong Wang, and Hao Chen. Boxinst: High-performance instance segmentation with box annotations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5443–5452, 2021. 7

[38] Long Wen, Yu Cheng, Yi Fang, and Xinyu Li. A comprehensive survey of oriented object detection in remote sensing images. *Expert Systems with Applications*, page 119960, 2023. 2

[39] Hongjin Wu, Ruoshan Lei, and Yibing Peng. Pcbnet: A lightweight convolutional neural network for defect inspection in surface mount technology. *IEEE Transactions on Instrumentation and Measurement*, 71:1–14, 2022. 1

[40] 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3974–3983, 2018. 1, 6, 8

[41] Xingxing Xie, Gong Cheng, Jiabao Wang, Xiwen Yao, and Junwei Han. Oriented r-cnn for object detection. In *IEEE/CVF International Conference on Computer Vision*, pages 3520–3529, 2021. 2

[42] Xue Yang and Junchi Yan. Arbitrary-oriented object detection with circular smooth label. In *European Conference on Computer Vision*, pages 677–694, 2020. 2

[43] Xue Yang and Junchi Yan. On the arbitrary-oriented object detection: Classification based approaches revisited. *International Journal of Computer Vision*, 130:1340–1365, 2022. 1

[44] Xue Yang, Hao Sun, Kun Fu, Jirui Yang, Xian Sun, Menglong Yan, and Zhi Guo. Automatic ship detection in remote sensing images from google earth of complex scenes based on multiscale rotation dense feature pyramid networks. *Remote Sensing*, 10(1), 2018. 1

[45] 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 *IEEE/CVF International Conference on Computer Vision*, pages 8231–8240, 2019. 2

[46] Xue Yang, Liping Hou, Yue Zhou, Wentao Wang, and Junchi Yan. Dense label encoding for boundary discontinuity free rotation detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15814–15824, 2021. 2

[47] Xue Yang, Junchi Yan, Ziming Feng, and Tao He. R3det: Refined single-stage detector with feature refinement for rotating object. In *AAAI Conference on Artificial Intelligence*, pages 3163–3171, 2021. 2

[48] Xue Yang, Junchi Yan, Ming Qi, Wentao Wang, Xiaopeng Zhang, and Tian Qi. Rethinking rotated object detection with gaussian wasserstein distance loss. In *International Conference on Machine Learning*, pages 11830–11841, 2021. 2, 7

[49] Xue Yang, Xiaojiang Yang, Jirui Yang, Qi Ming, Wentao Wang, Qi Tian, and Junchi Yan. Learning high-precision bounding box for rotated object detection via kullback-leibler divergence. In *Advances in Neural Information Processing Systems*, pages 18381–18394, 2021. 2

[50] Xue Yang, Gefan Zhang, Wentong Li, Xuehui Wang, Yue Zhou, and Junchi Yan. H2rbox: Horizontal box annotation is all you need for oriented object detection. *International Conference on Learning Representations*, 2023. 1, 2, 3, 7

[51] Xue Yang, Gefan Zhang, Xiaojiang Yang, Yue Zhou, Wentao Wang, Jin Tang, Tao He, and Junchi Yan. Detecting rotated objects as gaussian distributions and its 3-d generalization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):4335–4354, 2023. 2

[52] Xue Yang, Yue Zhou, Gefan Zhang, Jirui Yang, Wentao Wang, Junchi Yan, Xiaopeng Zhang, and Qi Tian. The kfiou loss for rotated object detection. In *International Conference on Learning Representations*, 2023. 2

[53] Ze Yang, Shaohui Liu, Han Hu, Liwei Wang, and Stephen Lin. Reppoints: Point set representation for object detection. In *IEEE/CVF International Conference on Computer Vision*, pages 9656–9665, 2019. 2, 7

[54] Xuehui Yu, Pengfei Chen, Di Wu, Najmul Hassan, Guorong Li, Junchi Yan, Humphrey Shi, Qixiang Ye, and Zhenjun Han. Object localization under single coarse point supervision. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4868–4877, 2022. 3

[55] Yi Yu and Feipeng Da. Phase-shifting coder: Predicting accurate orientation in oriented object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13354–13363, 2023. 2

[56] Yi Yu, Xue Yang, Qingyun Li, Yue Zhou, Gefan Zhang, Feipeng Da, and Junchi Yan. H2rbox-v2: Incorporating symmetry for boosting horizontal box supervised oriented object detection. In *Advances in Neural Information Processing Systems*, 2023. 1, 2, 3, 5, 6, 7, 8

[57] Shilong Zhang, Zhuoran Yu, Liyang Liu, Xinjiang Wang, Aojun Zhou, and Kai Chen. Group r-cnn for weakly semi-supervised object detection with points. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9407–9416, 2022. 3

[58] Zhong-Qiu Zhao, Peng Zheng, Shou-Tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE Transactions on Neural Networks and Learning Systems*, 30 (11):3212–3232, 2019. 2

[59] Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. East: An efficient and accurate scene text detector. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2642–2651, 2017. 1

[60] Yue Zhou, Xue Yang, Gefan Zhang, Jiabao Wang, Yanyi Liu, Liping Hou, Xue Jiang, Xingzhao Liu, Junchi Yan, Chengqi Lyu, Wenwei Zhang, and Kai Chen. Mmrotate: A rotated object detection benchmark using pytorch. In *ACM International Conference on Multimedia*, 2022. 6

[61] Tianyu Zhu, Bryce Ferenczi, Pulak Purkait, Tom Drummond, Hamid Rezatofighi, and Anton van den Hengel. Knowledge combination to learn rotated detection without rotated annotation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. 3, 7, 8