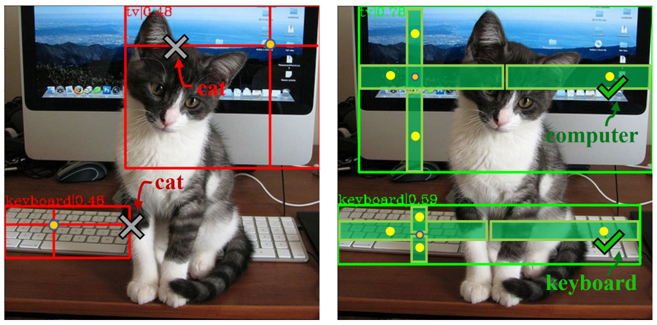
DE-CrossDet: Divisible and Extensible Crossline Representation for Object Detection

Hefei Mei, Hongliang Li, Heqian Qiu, Jianhua Cui, Longrong Yang
University of Electronic Science and Technology of China
hfmei@std.uestc.edu.cn, hlli@uestc.edu.cn, {hqqiu, jhcui, yanglr}@std.uestc.edu.cn

***Abstract*—Object detection aims to localize and classify objects. Suitable object representation plays an important role in accurate detection. Because a complete crossline inevitably passes through the noise of backgrounds or other objects, object features directly extracted by the whole crossline are often confused. In this paper, we present a new feature extraction method, DE-Crossline, which can enhance the original crossline representation to capture more accurate object information. Specifically, we divide the crossline into several segments, each of which extracts the maximum activation key point respectively to reduce the impact of noise mentioned above. Furthermore, considering various shapes and sizes of objects, we design a Deformable Width Extension Module to learn a suitable width of each crossline, so as to capture richer object information. Extensive experiments prove the effectiveness of our proposed method. The total performance of our proposed detector can reach 49.0% AP, using ResNet-101 as backbone on the MS-COCO dataset.**

***Index Terms*—Object detection, crossline, key points, better representation, deformable width**



(a) Crossline representation (b) DE-Crossline representation

Fig. 1. Different object features extraction methods by crossline. (a) uses the crossline to extract object features. (b) uses the DE-crossline to extract object features. The green crossline extractors indicate correct detection result, while the red extractors indicate wrong result. The continuous line on features of (a) represents feature extraction. The key points with maximum activation value on the segments of (b) represent features of the corresponding segments.

I. Introduction

Object detection is a fundamental component in computer vision, aiming to predict categories and bounding boxes based on object features. With the development of computer vision technology, image features extraction networks [1]–[5] have shown strong power. But besides that, detection tasks also need to extract object features by suitable representation cautiously for accurate detection. Current mainstream detectors mainly use anchor-based representation [6]–[13] or anchor-free representation [14]–[20] to extract object features. Both of them hope to construct object representation with more effective object information.

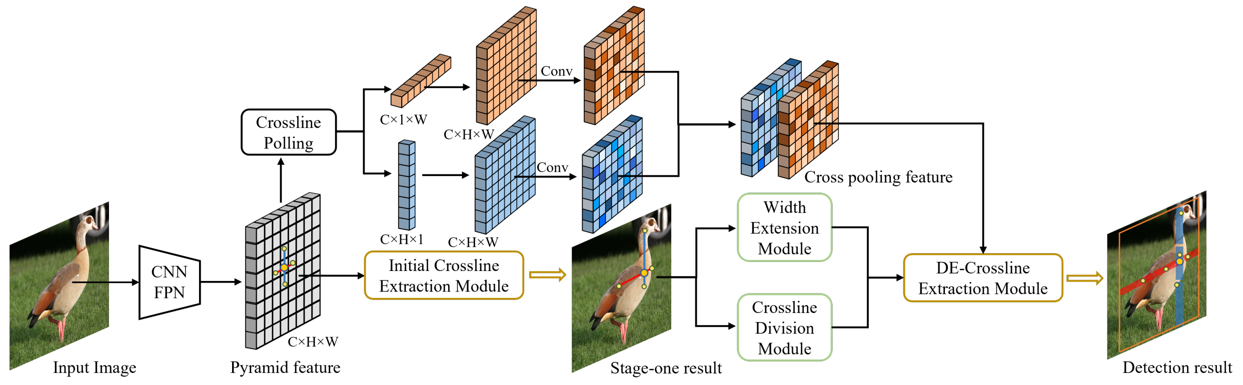
Anchor-based object detection networks [6]–[13] use anchor boxes to represent object features in the rectangular area. It can be divided into one-stage networks [12], [13] and two-stage networks [6]–[11], where the latter need to refine the detection results several times while the former need just once. As objects with irregularly shapes normally do not fill the entire anchor, the noise of backgrounds is also heavily contained in the anchor box, which may disturb prediction results.

Anchor-free object detection networks [14]–[20] achieve better performance after the emergence of feature pyramid networks (FPN) [21]. They generally use key points [14]–[19] to extract object features with less background noise. But these methods lose adjacent information of objects, which makes extracted object features not sufficient. CrossDet [20] uses crossline instead of key points to abstract continuous information. However, different objects and backgrounds are easy to overlap in natural scenes, a complete crossline may inevitably pass through wrong information, disturbing object detection results. As shown in Fig. 1(a), red horizontal line covering on the monitor and keyboard can not avoid passing through features of the cat, which causes the decrease of classification confidence and the position object to be incomplete.

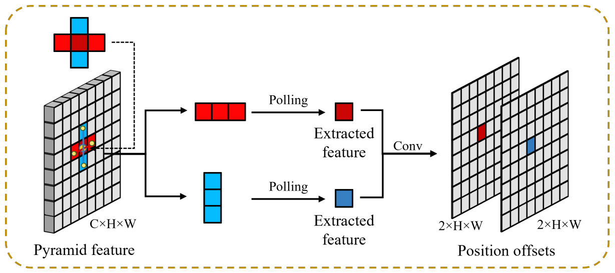
In this paper, we combine the advantages of key points and crossline representation, and propose DE-CrossDet using divisible and extensible crossline as object features extractor. Specifically, (1) we divide the crossline into several segments and use key points to represent features of segments. Then, we integrate the features of key points on segments along the x or y direction, aiming to obtain information that is not obscured by the noise of other objects or backgrounds. (2) Due to the diversity of object shapes, we design a Deformable Crossline Extension Module to extract object features with richer information. To make crossline width learnable, we design a two-stage network and predict width offsets of the crossline at the second stage. Deformable convolution [22] is also incorporated in our module to adapt to the shape diversity of objects. Illustrated in Fig. 1(b), key points on horizontal segments with suitable width can extract complete features of the monitor and the keyboard, so as to obtain higher classification confidence and the correct position.

The main contributions of our work are summarized as:

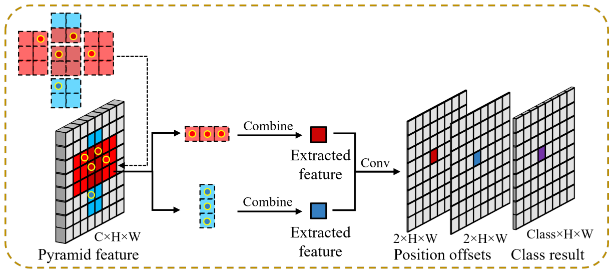
- We use key points on crossline segments to extract object features, which can reduce the influence of interfering



(a) The network architecture of our method



(b) Initial Crossline Extraction Module



(c) DE-Crossline Extraction Module

Fig. 2. Overview of our proposed DE-CrossDet. (a) provides an overall network architecture of our method, which shows one scale of FPN [21] feature maps. We regress the position of each initial crossline at the first stage. Based on the approximate crossline grown at previous stage, we extend its width adaptively to capture richer object information. In Crossline Division Module, we divide the crossline into several segments, then combine extracted features to reduce the interference caused by object occlusion. (b) shows the feature extraction process in Initial Crossline Extraction Module. $2 \times H \times W$ denotes the offsets of height and width. (c) shows how we get final results by DE-Crossline Extraction Module, where the Class means the number of categories.

objects or backgrounds.

- The width of crossline can be extended adaptively based on detecting objects, which extracts richer object information and adapts to the shape diversity of objects.
- The experiments on both PASCAL VOC dataset [23] and MS-COCO dataset [26] demonstrate the effectiveness of the detector we propose.

II. Proposed Method

Compared with CrossDet [20], the crossline in DE-CrossDet is divisible and extensible. We first place initial crossline on the pyramid feature to predict the coarse object position. The pyramid feature is also used to get crossline pooling feature, which can refine features in the second stage. Finally, we divide the coarse crossline and predict its deformable width, which is used for accurate detection in the second stage.

A. Initial Crossline

1) Crossline Extractor: In this paper, we treat an object as key points on several crossline segments with deformable width. The architecture of our proposed DE-CrossDet is illustrated in Fig. 2. Specifically, in the first stage, we place an initial crossline on each point of the pyramid feature $F \in \mathbb{R}^{C\times H\times W}$ to extract object features. The feature extraction module at the first stage is shown in Fig. 2(b), the horizontal line at center point (x_0, y_0) can be expressed as $H_{init} = (x_0 - 1, x_0 + 1, y_0)$ while the vertical line is $V_{init} = (x_0, y_0 - 1, y_0 + 1)$. Let $F(x, y)$ be the activation value on features, horizontal and vertical pooling results E_{row} and E_{col} can be calculated as:

$$E_{row} = \max_{x\in[x_0-1,x_0+1]} F(x, y_0) \tag{1}$$

$$E_{col} = \max_{y\in[y_0-1,y_0+1]} F(x_0, y) \tag{2}$$

2) Crossline Pooling: When the initial crossline $H_{init} = (x_0 - 1, x_0 + 1, y_0)$ and $V_{init} = (x_0, y_0 - 1, y_0 + 1)$ of each center point (x_0, y_0) grows on features, it has limitations in only extracting the local information of the corresponding cross. In order to extract more reliable regression information, we compress the pyramid feature to $\mathbb{R}^{C\times 1\times W}$ horizontally and $\mathbb{R}^{C\times H\times 1}$ vertically by crossline pooling. To improve feature information in the second stage, these two pooling features are extended to the same size as the pyramid feature F and then combined with it by a 3×3 convolution layer. The Cross pooling feature $F_{row} \in \mathbb{R}^{C\times H\times W}$ and $F_{col} \in \mathbb{R}^{C\times H\times W}$ will be used to extract refine object features in the second stage.

B. DE-Crossline

1) Crossline Division Module: To overcome the confusion of features caused by the noise of backgrounds or other objects, we divide a crossline into n segments. For each crossline in the second stage, we can express horizontal line and vertical line on the center point (x_0, y_0) as $H_{refine} = (x_1, x_2, y_0)$ and

$V_{refine} = (x_0, y_1, y_2)$. Different from the crossline extractor, which use all crossline to extract feature, we calculate the segment expression as:

$$\begin{aligned} H_{refine}^{(k)} &= (x_1^{(k)}, x_2^{(k)}, y_0) \\ &= (x_1 + \frac{k-1}{n}(x_2 - x_1), x_1 + \frac{k}{n}(x_2 - x_1), y_0) \end{aligned} \tag{3}$$

$$\begin{aligned} V_{refine}^{(k)} &= (x_0, y_1^{(k)}, y_2^{(k)}) \\ &= (x_0, y_1 + \frac{k-1}{n}(y_2 - y_1), y_1 + \frac{k}{n}(y_2 - y_1)) \end{aligned} \tag{4}$$

where k is the number of segments from the starting point.

2) Deformable Width Extension Module: The crossline with extended width can extract richer object information. Considering the rigidity of the line and the diversity of the shape and size of the object, we predict deformable width adapted to different shapes of objects for accurate detection.

To capture fine grained object information, we incorporate deformable convolution [22] into our module. As shown in Fig. 3, the extracted features $E \in \mathbb{R}^{C\times H\times W}$ by crossline are the same size as pyramid feature. On this foundation, we combine these features to get feature $F_{com} \in \mathbb{R}^{2C\times H\times W}$, which contains the crossline information and object information. Then we get $offsets \in \mathbb{R}^{2n\times H\times W}$ by a 3×3 convolution layer, where $n = 9$ represents the number of key points in the deformable convolution layer. At last, we can obtain the deformable convolution features $F_{dcn} \in \mathbb{R}^{C\times H\times W}$, which is used to predict width offsets wid_off by a convolution layer. The width of lines can be formulated as:

$$width^{(2)} = width^{(1)} \times e^{wid\_off} \tag{5}$$

where $width^{(2)}$ is width of the crossline in the second stage while $width^{(1)}$ is width of the crossline in the first stage.

3) Crossline features integration: In the second stage, we extract the improved feature described above by dividing and extending crossline. Each crossline segments also extract max-pooling features as mentioned in (1) and (2). The features extracted by the segments along the x or y direction as $E^{(1)}$, $E^{(2)}$ and $E^{(n)}$ can be combined as follows:

$$E_{row} = \mathcal{C}_{3\times3}(E_{row}^{(1)} + E_{row}^{(2)} + \cdots + E_{row}^{(n)}) \tag{6}$$

$$E_{col} = \mathcal{C}_{3\times3}(E_{col}^{(1)} + E_{col}^{(2)} + \cdots + E_{col}^{(n)}) \tag{7}$$

where $\mathcal{C}_{3\times3}(\cdot)$ represents the 3×3 convolution and n is the number of crossline segments.

III. Experiments

Baseline. Different from CrossDet [20], which uses the summation features of Crossline pooling in two directions, we use Crossline pooling in each direction separately when getting position offsets in the second stage. In order to make a fair comparison of the methods used in our paper, we remove the Crossline Pooling module of both two detectors and then present the ablation experiment results. Apart from it, we also compare the overall performance with CrossDet [20].

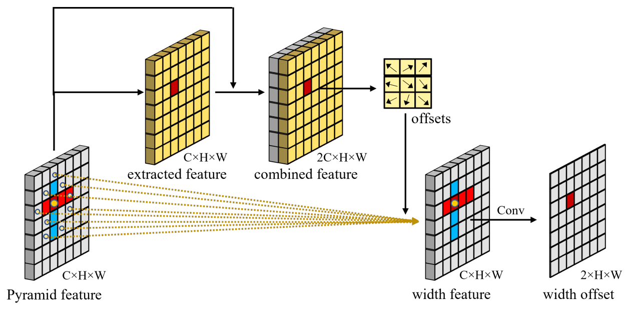


Fig. 3. Structure of Width Extension Module. The features extracted by crossline without width will be combined with pyramid feature to guide the offsets of deformable convolution.

Dataset. We evaluate performance of each module on the PASCAL VOC dataset [23], which contains 20 categories and 21k images. Among them, we choose VOC2007 *trainval* and VOC2012 *trainval* with 16551 images to train our detector and VOC2007 *test* with 4952 image for validation. Moreover, we evaluate DE-CrossDet on the MS-COCO dataset [26], which contains 118k images for training, 5k images for validation and 20k images for testing.

Implementation Details. In this paper, we train our detector using stochastic gradient descent (SGD) with 4 images per batch (2 GPUs, 2 images per GPU) on the PASCAL VOC dataset [23] and 8 images per batch (8 GPUs, 1 images per GPU) on the MS-COCO dataset [26]. The backbone is initialized through ImageNet [24] pre-trained model. Following [20], the IoU threshold $\sigma = 0.6$ of NMS and the loss weights of two stages is $\lambda^1 = 1$ and $\lambda^2 = 2$.

A. Comparison with CrossDet

We compare our method with other state-of-the-art detectors using ResNet-50 and ResNet-101 [3] as backbone on the PASCAL VOC dataset [23] in Table I. DE-CrossDet can achieve 51.7% AP using ResNet-50 as backbone and 53.4% AP using ResNet-101 as backbone. As shown in Table II, DE-CrossDet with ResNet-101-DCN achieves 49.0% AP with multi-training and multi-testing, surpassing CrossDet by 0.6% on the MS-COCO dataset [26]. We also show some detection results with DE-crossline in Fig. 4, where DE-Crossline can adjust the width according to the shape and size of objects.

B. Ablation Study

1) Number of divided segments: As mentioned before, we always divide crosslines with crossline segments in network learning. Table III compares detector performance with the different number of crossline segments, where N indicates the number of segments we use. When the number of line segments increases, the performance of the detector will begin to decline from 49.7% to 49.5%. It is not that the more segments are divided, the more accurate detector is, which can justify the rationality of fixed and small quantities of segments.

2) Divisible and Extensible Crossline: To prove the effectiveness of the Division and Extension Module we proposed, we add each of our modules to CrossDet [20] baseline respectively. Table IV shows the benefit of Crossline Division

TABLE I
COMPARISON OF OUR METHOD WITH OTHER STATE-OF-THE-ART DETECTORS ON PASCAL VOC [23].

Method	Backbone	Representation Way	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster R-CNN [8]	ResNet-50	Anchor-based	46.5	75.8	50.5	11.9	33.8	55.6
RetinaNet [12]	ResNet-50	Anchor-based	47.2	73.2	50.2	12.1	33.0	56.7
Reppoints [19]	ResNet-50	Point-based	45.9	73.5	48.8	11.6	32.2	55.0
FCOS-imprv [17]	ResNet-50	Point-based	47.6	72.2	51.1	11.7	31.8	57.7
ATSS [25]	ResNet-50	Point-based	49.3	73.8	53.6	13.5	36.2	58.8
CrossDet [20]	ResNet-50	Crossline-based	50.9	75.4	55.2	15.0	36.6	60.8
DE-CrossDet	ResNet-50	Crossline-based	**51.7**	**76.2**	**56.5**	**15.1**	**36.6**	**61.8**
Faster R-CNN [8]	ResNet-101	Anchor-based	48.8	75.5	53.5	10.7	35.9	58.3
RetinaNet [12]	ResNet-101	Anchor-based	49.5	74.1	53.0	12.4	35.5	59.3
RepPoints [19]	ResNet-101	Point-based	47.5	74.2	51.3	11.4	33.9	56.9
FCOS-imprv [17]	ResNet-101	Point-based	49.7	73.4	54.3	12.4	34.7	60.1
ATSS [25]	ResNet-101	Point-based	51.7	75.2	57.1	12.7	37.1	61.9
CrossDet [20]	ResNet-101	Crossline-based	52.8	76.9	57.9	13.7	38.0	63.4
DE-CrossDet	ResNet-101	Crossline-based	**53.4**	**77.4**	**58.7**	**14.3**	**38.0**	**63.7**

TABLE II
COMPARISON OF OUR METHOD WITH OTHER STATE-OF-THE-ART DETECTORS ON MS-COCO [26].

Method	Backbone	Representation Way	Epoch	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster R-CNN w. FPN [8]	ResNet-101	Anchor-based	24	36.2	59.1	39.0	18.2	39.0	48.2
RetinaNet [12]	ResNet-101	Anchor-based	-	39.1	59.1	42.3	21.8	42.7	50.2
Cascade R-CNN [10]	ResNet-101	Anchor-based	18	42.8	62.1	46.3	23.7	45.5	55.2
SABL [27]	ResNet-101	Anchor-based	24	43.3	60.9	46.2	23.8	46.5	55.7
ExtremeNet † [29]	Hourglass-104	Point-based	200	40.2	55.5	43.2	20.4	43.2	53.1
CornerNet † [14]	Hourglass-104	Point-based	200	40.5	56.5	43.1	19.4	42.7	53.9
CenterNet † [16]	Hourglass-104	Point-based	190	44.9	62.4	48.1	25.6	47.4	57.4
FCOS-imprv † [17]	ResNet-101	Point-based	24	43.0	61.7	46.3	26.0	46.8	55.0
RepPoints † [19]	ResNet-101-DCN	Point-based	24	45.0	66.1	49.0	26.6	48.6	57.5
ATSS † [25]	ResNet-101-DCN	Point-based	24	46.3	64.7	50.4	27.7	49.8	58.4
BorderDet † [28]	ResNet-101-DCN	Point-based	24	47.2	66.1	51.0	28.1	50.2	59.9
CrossDet [20]	ResNet-50	Crossline-based	12	41.1	60.1	44.7	24.4	43.8	51.0
CrossDet [20]	ResNet-101	Crossline-based	12	42.8	61.9	46.7	25.1	45.7	53.5
CrossDet † [20]	ResNet-101-DCN	Crossline-based	24	47.4	65.9	52.3	29.5	50.2	58.7
CrossDet †† [20]	ResNet-101-DCN	Crossline-based	24	48.4	66.4	**54.1**	**32.0**	50.6	59.0
DE-CrossDet	ResNet-50	Crossline-based	12	41.8	60.6	45.5	24.4	44.8	52.3
DE-CrossDet	ResNet-101	Crossline-based	12	43.5	62.6	47.3	25.2	46.8	54.7
DE-CrossDet †	ResNet-101-DCN	Crossline-based	24	48.1	66.9	52.5	29.6	51.5	60.5
DE-CrossDet ††	ResNet-101-DCN	Crossline-based	24	**49.0**	**67.5**	53.8	30.6	**52.4**	**61.9**

The symbol † indicates multi-scale training while symbol †† indicates multi-scale training and testing.

TABLE III
DIFFERENT NUMBER OF DIVIDED SEGMENTS.

N of segments	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
4	49.5	74.0	54.1	13.4	34.5	59.5
3	49.6	74.0	53.8	13.1	**35.3**	59.4
2	**49.7**	**74.2**	**54.2**	**13.6**	35.0	**59.5**
1	48.9	73.8	53.1	12.8	34.9	58.7

TABLE IV
ANALYSIS OF OUR CROSSLINE PROCESSING MODULE.

Div.	Ext.	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
		48.9	73.8	53.1	12.8	34.9	58.7
✓		49.7	74.2	54.2	**13.6**	35.0	59.5
	✓	49.3	74.3	53.2	13.0	34.9	59.2
✓	✓	**50.4**	**74.8**	**55.0**	12.9	**36.0**	**60.4**

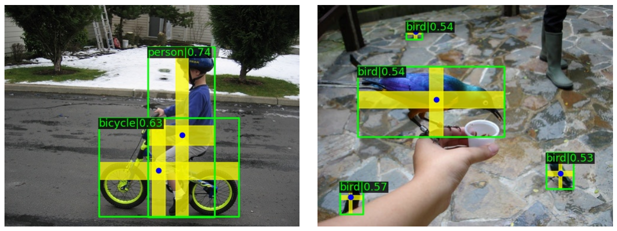


Fig. 4. Detection results of DE-CrossDet. The blue dot is the intersection of DE-Crossline. The bounding box is obtained by appropriate DE-Crossline.

and Deformable Extension Module. CrossDet [20] with our Division Module can outperform baseline by 0.8% while it with our Extension Module can outperform baseline by 0.4%. Our overall detector based on CrossDet [20] achieves 50.4% AP, outperforming the baseline by 1.5%.

IV. CONCLUSION

In this work, we propose two effective modules to enhance the representation of crossline. The deformable width of crossline for different object shapes can extract object features with richer information. The key points on divisible crossline segments can reduce detection noise, so as to obtain better detection performance. Based on our new representation method, we develop a detector called DE-CrossDet that achieves competitive performance without bells and whistles.

ACKNOWLEDGMENT

This work was supported in part by National Natural Science Foundation of China under Grant 61831005, Grant 62071086 and Sichuan Science and Technology Program under Grant 2021YFG0296.

References

[1] K. Simonyan, A. Zisserman , "Very deep convolutional networks for large-scale image recognition," in *2015 International Conference on Learning Representations(ICLR)*, 2015, pp. 1-14.

[2] C. Szegedy , W. Liu , Y. Jia, "Going deeper with convolutions," in *2015 Computer Vision and Pattern Recognition(CVPR)*, 2015, pp. 1-9.

[3] K. He, X. Zhang, S. Ren, "Deep Residual Learning for Image Recognition," in *2016 Computer Vision and Pattern Recognition(CVPR)*, 2016, pp. 770-778.

[4] S. H. Gao, M. M. Cheng, K. Zhao, "Res2Net: A New Multi-scale Backbone Architecture," in IEEE transactions on pattern analysis and machine intelligence, 2019, pp.652-662.

[5] S. Xie, R. Girshick, P. Dollár, and Z. Tu, "Aggregated residual transformations for deep neural networks," in *2017 Computer Vision and Pattern Recognition(CVPR)*, 2017, pp.1492-1500.

[6] R. Girshick, J. Donahue, T. Darrell, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *2014 Computer Vision and Pattern Recognition(CVPR)*, 2014, pp. 580-587.

[7] R. Girshick, "Fast rcnn," in *2015 International Conference on Computer Vision(ICCV)*, 2015, pp. 1440-1448.

[8] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in neural information processing systems, 2015, pp. 1137-1149.

[9] Y. Guo , X. Guo, Z. Jiang, "Cascaded convolutional neural networks for object detection," in *2017 Visual Communications and Image Processing (VCIP)*, 2017, pp. 1-4.

[10] Z. Cai and N. Vasconcelos, "Cascade r-cnn: Delving into high quality object detection," in *2018 Computer Vision and Pattern Recognition(CVPR)*, 2018, pp. 6154-6162.

[11] H. Qiu, H. Li, Q. Wu, "Offset bin classification network for accurate object detection," in *2020 Computer Vision and Pattern Recognition(CVPR)*, 2020, pp. 13188-13197.

[12] TY. Lin, P. Goyal, R. Girshick, "Focal loss for dense object detection," in *2017 International Conference on Computer Vision(ICCV)*, 2017, pp. 2980-2988.

[13] W. Liu, D. Anguelov, D. Erhan, "Ssd: Single shot multibox detector," in *2016 European Conference on Computer Vision(ECCV)*, 2016, pp. 21-37.

[14] H. Law, J. Deng, "Cornernet: Detecting objects as paired keypoints," in *2018 European Conference on Computer Vision(ECCV)*, 2018, pp. 734-750.

[15] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *2016 Computer Vision and Pattern Recognition(CVPR)*, 2016, pp. 779-788.

[16] K. Duan, S. Bai, L. Xie, "Centernet: Keypoint triplets for object detection," in *2019 International Conference on Computer Vision(ICCV)*, 2019, pp. 6569-6578.

[17] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: Fully convolutional one-stage object detection," in *2019 International Conference on Computer Vision(ICCV)*, 2019, pp. 9627-9636.

[18] X. Zhou, D. Wang, P. Krähenbühl, "Objects as points," in arXiv preprint, 2019, arXiv:1904.07850.

[19] Z. Yang, S. Liu, H. Hu, "Reppoints: Point set representation for object detection," in *2019 International Conference on Computer Vision(ICCV)*, 2019, pp. 9657-9666.

[20] H. Qiu, H. Li, Q. Wu, and J. Cui, "CrossDet: Crossline Representation for Object Detection," in *2021 International Conference on Computer Vision(ICCV)*, 2021, pp. 3195-3204.

[21] T. Y. Lin, P. Dollár, R. Girshick, and K. He, "Feature pyramid networks for object detection," in *2017 Computer Vision and Pattern Recognition(CVPR)*, 2017, pp. 2117-2125.

[22] J. Dai, H. Qi, Y. Li, "Deformable convolutional networks," in *2017 International Conference on Computer Vision(ICCV)*, 2017, pp. 764-773.

[23] M. Everingham, L. Van Gool, C. K. Williams, "The pascal visual object classes (voc) challenge," in *International Journal of Computer Vision*, 2010, pp. 303-338.

[24] O. Russakovsky , J. Deng , H. Su, "Imagenet large scale visual recognition challenge," in *International Journal of Computer Vision*, 2015, pp. 211-252.

[25] S. Zhang, C. Chi, Y. Yao, and Z. Lei, "Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection," in *2020 Computer Vision and Pattern Recognition(CVPR)*, 2020, pp. 9759-9768.

[26] T. Y. Lin, M. Maire, S. Belongie, and J. Hays, "Microsoft coco: Common objects in context," in *European conference on computer vision*, 2014, pp. 740-755.

[27] J. Wang, W. Zhang, Y. Cao, "Side-aware boundary localization for more precise object detection," in *European Conference on Computer Vision*, 2020, pp. 403-419.

[28] H. Qiu, Y. Ma, Z. Li, "Borderdet: Border feature for dense object detection," in *European Conference on Computer Vision*, 2020, pp. 549-564.

[29] X. Zhou, J. Zhuo, P. Krahenbuhl, "Bottom-up object detection by grouping extreme and center points," in *2019 Computer Vision and Pattern Recognition(CVPR)*, 2019, pp. 850-859.