Project 6: Oriented Object Detection Model for Aerial Images

Hrishikesh Rana, Divya Patel, Aditya Chaudhari, Preksha Morbia
Ahmedabad University

Abstract

Oriented Object Detection is challenging, as the objects in aerial images are displayed in arbitrary directions. Current methods rely on two-stage anchor-based detectors, which typically suffer from severe imbalances between positive and negative anchor boxes. This model detects center key points and regresses box boundary-aware vectors (BBAVectors) to capture oriented bounding boxes across the cartesian plane. This approach outperforms two-stage anchor-based detectors by directly predicting box parameters.

1. Keywords

Oriented object detection, Box boundary-aware vectors, Object bounding boxes, Key point-based detection, Anchor-free detection, Single-stage detector, Rotational object detection, Remote sensing, Arbitrary orientation, Center key point prediction, Heatmap, Offset prediction, Orientation classification, Corner case handling, DOTA dataset, Convolutional neural networks, Aerial imagery analysis

2. Introduction

Object detection in aerial images is crucial. However, it is difficult to do so due to the variance in textures, scales, complex backgrounds, and arbitrary object orientations. Horizontal bounding boxes don't align well with objects, prompting a shift to oriented bounding boxes. Current methods use two-stage anchor-based detectors, with the first stage proposing regions via anchor boxes and the second refining them. However, these methods need more design complexity, imbalance issues, and computational costs. Zhou's CenterNet proposes directly regressing width and height but faces challenges in jointly learning parameters for arbitrarily oriented objects.

3. Methodology

The paper uses pre-trained ResNet101 Conv1-5 as the foundation for this neural network that helps extract features from input images for accurate predictions.

First, we up-sample the image by increasing the size of extracted features to make it four times smaller, which is done to retain finer details.

The up-sampled feature map is refined through a 3 X 3 Conv layer. This refined feature map is concatenated with the shallow layer, followed by a 1x1 convolutional layer to refine channel-wise features.

The Output feature map is transformed into four branches:

I. **Heatmaps:** Heat maps are used here to detect oriented objects' center points. The heatmap P is a tensor of shape. Here, K is the number of channels, with each corresponding to one object category. Each of these channels is passed through a sigmoid function. The predicted heatmap value at a particular centre point is the confidence of the object detection. Groundtruth heatmaps

p̂ are formed by placing 2D Gaussians around object centre points, using a Gaussian function with size-adaptive standard deviation σ, resulting in $K \times H_s \times W_s$ tensor.

Training loss: Positive samples are only cantered points c, while all other points, including those within Gaussian bumps, are harmful. A variant focal loss is used to train the heatmap to handle imbalance issues.

$$L_h = -\frac{1}{N}\sum_i \begin{cases} (1-p_i)^{\alpha}\log(p_i) \\ (1-\hat{p}_i)^{\beta} p_i^{\alpha}\log(1-p_i) \end{cases}$$

Where $\hat{p}$ and *prefer to the ground truth and the predicted heatmap values, I indexes the pixel locations on the* Type equation here.*feature map, N is the number of objects, and* α and β are the hyperparameters controlling each point's contribution.

II. **Offset:** During inference, peak points are extracted from the predicted heatmap P as object centre point locations. These centre points c are integers. However, downsampling a point from the input image to the output heatmap results in a floating-point number. An offset map is predicted to compensate for the difference between the quantified floating center point (generated from a downscaling image) and the integer center point (generated from heat maps). This is done using **smooth L1 loss** with ground-truth offsets to ensure correct localisation.

Training loss:

$$L_o = \frac{1}{N}\sum_{k=1}^{N} \text{Smooth}_{L_1}(\mathbf{o}_k - \hat{\mathbf{o}}_k),$$

Where N is the total number of objects, oˆ refers to the ground-truth offsets, and k indexes the objects. **The smooth L1 loss** can be expressed as:

$$\text{Smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise.} \end{cases}$$

III. **Box-Parameters:** The parameters for box boundary aware (BBA) vectors contain the top t, correct r, bottom b, and left l vectors, which are from the center points of the objects. These four vectors are distributed in four quadrants of the cartesian coordinate system. This helps enhance mutual information sharing.

Training loss: It is trained via **smooth L1 loss** to regress BBA vectors and external size parameters for improved results.

$$L_b = \frac{1}{N}\sum_{k=1}^{N} \text{Smooth}_{L_1}(\mathbf{b}_k - \hat{\mathbf{b}}_k),$$

Where b and $\hat{b}$ are the predicted and ground-truth box parameters, respectively.

IV. **Orientation:** Objects are categorized into two types of bounding boxes, HBB and RBB, which help address detection failures in corner cases by transforming them into horizontal ones that are easy to deal with. We define the orientation map as asterisk operator, cap W over denominator $\alpha \in \mathbb{R}^{1 * \frac{H}{s} * \frac{W}{s}}$. A sigmoid function finally processes the output map. To create the ground truth of the orientation class, we define:

Method	mAP	Plane	BD	Bridge	GTF	SV	LV	Ship	TC	BC	ST	SBF	RA	Harbor	SP	HC
YOLOv2 [18]	25.49	52.75	24.24	10.6	35.5	14.36	2.41	7.37	51.79	43.98	31.35	22.3	36.68	14.61	22.55	11.89
FR-O [22]	54.13	79.42	77.13	17.7	64.05	35.3	38.02	37.16	89.41	69.64	59.28	50.3	52.91	47.89	47.4	46.3
R-DFPN [24]	57.94	80.92	65.82	33.77	58.94	55.77	50.94	54.78	90.33	66.34	68.66	48.73	51.76	55.10	51.32	35.88
R^2CNN [7]	60.67	80.94	65.75	35.34	67.44	59.92	50.91	55.81	90.67	66.92	72.39	55.06	52.23	55.14	53.35	48.22
Yang *et al.* [25]	62.29	81.25	71.41	36.53	67.44	61.16	50.91	56.60	90.67	68.09	72.39	55.06	55.60	62.44	53.35	51.47
ICN [1]	68.16	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
ROI Trans. [2]	67.74	88.53	77.91	37.63	**74.08**	66.53	62.97	66.57	90.5	79.46	76.75	**59.04**	56.73	62.54	61.29	55.56
ROI Trans.+FPN [2]	69.56	**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
BBAVectors+r	71.61	88.54	76.72	49.67	65.22	75.58	**80.28**	87.18	90.62	**84.94**	**84.89**	47.17	60.59	65.31	63.91	53.52
BBAVectors+rh	**72.32**	88.35	**79.96**	**50.69**	62.18	**78.43**	78.98	**87.94**	**90.85**	83.58	84.35	54.13	60.24	65.22	**64.28**	**55.70**
BBAVectors+rh^*	**75.36**	**88.63**	**84.06**	**52.13**	69.56	**78.26**	**80.40**	**88.06**	**90.87**	**87.23**	**86.39**	56.11	**65.62**	**67.10**	**72.08**	**63.96**

Table 1. Detection results on the testing set of DOTA-v1.0. The performances are evaluated through the online server. Symbol $*$ shows the result with a larger training batch size (i.e., 48 on 4 Quadro RTX 6000 GPUs). Red and Blue colors label the best and second best detection results in each column.

mAp	Two-Wheeler	Three-Wheeler	Bicycle	Pedestrian	People	Cars	Heavy Vehicles
84.71	97.18	94.07	95.66	35.29	8.00	94.69	96.07

Table 2. Our Results

$$\hat{\alpha} = \begin{cases} 1 \text{ (RBB)} & \text{IOU(OBB, HBB)} < 0.95 \\ 0 \text{ (HBB)} & \text{otherwise,} \end{cases}$$

IOU is the intersection-over-union between the oriented bounding box (OBB) and the horizontal bounding box (HBB).

Training loss: The orientation class is trained with the **binary cross-entropy loss**:

$$L_\alpha = -\frac{1}{N}\sum_{i}^{N}(\hat{\alpha}_i \log(\alpha_i) + (1-\hat{\alpha}_i)\log(1-\alpha_i)),$$

Where α and are the predicted and the ground-truth orientation classes, respectively.

Result Image 1

Result Image 2

4. Results:

With the orientation classification and additional external Oriented Bounding Box (OBB) size parameters, our method achieves 72.32% accuracy. With the larger training batch size, it achieves 75.36% accuracy.
Refer to Table 1.

We divided the dataset into seven classes less than the DOTA dataset, but the number of objects differed for different courses. Thus, to tackle

imbalance, we added weights for each class. This was the main reason we got better precision than the original model. We achieved accuracy of 84.71%. The detection speed of the model is 11 FPS.
Refer to Table 2.
Results are show visually in images.
Refer to Result Image 1 and Result Image 2.

5. Discussion:

- Firstly, we had dependency issues because of which the model was not running on our system. So, we resolved the issue by creating a virtual environment.
- As the provided dataset of AU drone was not in the same format as required. So, we converted the dataset into DOTA dataset and split the dataset.
- After splitting, for one video we were getting around 28,000 images. It required 24 hours to run these 28,000 images for just 1 epoc. So, we ran statistics to check the number of objects per frame. From this, we took those frames data whose number of objects were between 15-20 which was around 6000 frames. This was then split for training and testing. Out of these, around 4000 frames were used for training.
- Before there were 10 classes in the dataset. So, we did changes in the code for the output of these 10 classes.
- Even then we were getting very few objects for classes like bus, van. So, we decreased the number of classes by clubbing few of them together and made it 7 classes.
- Then, for improving output we added weights to different classes according to their variation. For example, there were more Two-wheelers than there were cars so we added weights accordingly so it causes no problem while training.
- Lastly. we observed and studied the results and checked that object is properly fitting in the orientation detected.
- Also, for using fast system we used a 16GB GPU in which we were getting faster results.

6. Conclusion

The proposed oriented object detection method using box boundary-aware vectors and center points detection is single-stage and anchor box-free, outperforming baseline methods on the DOTA dataset. Its simplicity and effectiveness make it a promising approach for future-oriented object detection tasks. We ran this model on our dataset. For improvement, still, people and pedestrians are not properly getting detected for which we balanced classes.

7. Reference:

- J. Yi, P. Wu, B. Liu, Q. Huang, H. Qu, and D. Metaxas, "Oriented Object Detection in Aerial Images with Box Boundary-Aware Vectors." Accessed: Mar. 16, 2024. [Online]. Available: https://arxiv.org/pdf/2008.07043v2.pdf

Ultralytics, "Home," Ultralytics YOLOv8 Docs, https://docs.ultralytics.com/