Measuring the Impact of Rotation Equivariance on Aerial Object Detection

Supplementary Material

7. An Example Where Downsampling Breaks Rotation Equivariance

In Sec. 3.2, we describe how conventional downsampling layers can break strict rotation equivariance. This occurs because the center sampling points of the convolution kernels do not match before and after rotation on even-sized feature maps, as illustrated in the left part of Fig. 7. The grid illustrates the padded feature map and the orange-highlighted pixel represents the convolution kernel's center sampling point.

For example, consider a clockwise 90-degree rotation. Define a grayscale image as a matrix, assuming the image has a size of $2n \times 2n$. Let x represent the column index and y represent the row index, $1 \leq x, y \leq 2n, x, y \in \mathbb{Z}$. When performing downsampling on the image using a convolutional kernel with a stride of 2, the coordinates of the sampling points are as follows:

$$(x, y) = (2i+1, 2j+1), i, j \in [0, n-1]. \quad (6)$$

If f represents the grayscale value of a pixel in the image at a certain coordinate, after rotating the image 90 degrees clockwise, the relationship between the coordinates (x', y') of the rotated pixel and the original pixel coordinates is given by:

$$f(x', y') = f(2n - y + 1, x). \quad (7)$$

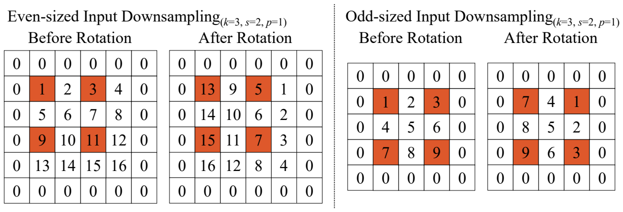
Similarly, after rotating the image 90 degrees clockwise, if a stride-2 convolution kernel is used for downsampling, the coordinates of the sampling points in the rotated image are given by:

$$(x', y') = (2i+1, 2j+1), i, j \in [0, n-1]. \quad (8)$$

By substituting the corresponding relationships from Eq. (7) into Eq. (8), the coordinates of the sampling pixel points in the rotated image can be expressed in terms of their original coordinates as:

$$(x, y) = (2j+1, 2n-2i), i, j \in [0, n-1]. \quad (9)$$

From the differences between Eq. (6) and Eq. (9), it can be observed that the row indices of the sampling points before rotation are all odd, while the row indices of the sampling points after rotation are all even. The difference in sampling points leads to varying convolution results for the same convolution kernel on the same feature map, which breaks strict rotation equivariance.



Even-sized Input Downsampling$_{(k=3,\ s=2,\ p=1)}$

Before Rotation

0	0	0	0	0	0
0	**1**	2	**3**	4	0
0	5	6	7	8	0
0	**9**	10	**11**	12	0
0	13	14	15	16	0
0	0	0	0	0	0

After Rotation

0	0	0	0	0	0
0	**13**	9	**5**	1	0
0	14	10	6	2	0
0	**15**	11	**7**	3	0
0	16	12	8	4	0
0	0	0	0	0	0

Odd-sized Input Downsampling$_{(k=3,\ s=2,\ p=1)}$

Before Rotation

0	0	0	0	0
0	**1**	2	**3**	0
0	4	5	6	0
0	**7**	8	**9**	0
0	0	0	0	0

After Rotation

0	0	0	0	0
0	**7**	4	**1**	0
0	8	5	2	0
0	**9**	6	**3**	0
0	0	0	0	0

Figure 7. Breaking and maintaining strict rotation equivariance.

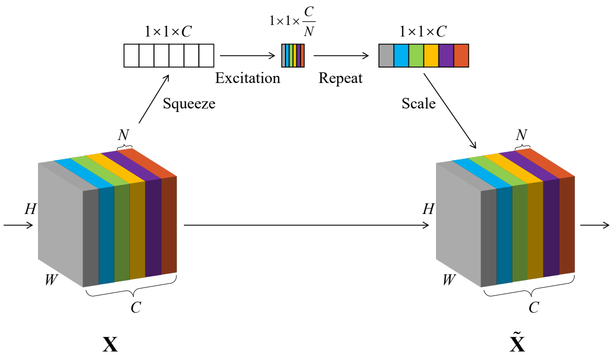


Figure 8. Rotation-Equivariant Channel Attention Mechanism.

To address this issue, we ensure strict rotation equivariance by using odd-sized inputs for all 2x downsampling layers, which preserves the alignment of convolution kernel center points before and after rotation, as illustrated in the right part of Fig. 7. The theoretical proof, as shown in Section 3.2 of [16], establishes that, to maintain strict rotation equivariance, the input size i, kernel size k, and stride s of a downsampling layer—regardless of whether it is a convolutional layer or a pooling layer—must satisfy the condition $(i-k) \bmod s = 0$. Notably, the downsampling method proposed in this study adheres to this condition.

8. The Further Details of MessDet

This paper introduces the rotation-equivariant channel attention (RE-CA), enabling rotation-equivariant networks to be implemented with more advanced network structures. The mathematical formulation of RE-CA is provided in Sec. 4.2, and its schematic diagram is shown in Fig. 8. In the figure, "Squeeze" refers to the global average pooling process, and "Excitation" refers to the fully connected layer and activation function. After obtaining the C/N-dimensional weight vector, each component of the vector is repeated N times to obtain the N-dimensional weight vector that preserves rotation equivariance.

Method	#P↓	mAP↑	PL	BD	BR	GTF	SV	LV	SH	TC	BC	ST	SBF	RA	HA	SP	HC
Two-stage																	
SCRDet[56]	41.9M	72.61	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
CSL[55]	37.4M	76.17	**90.25**	85.53	54.64	75.31	70.44	73.51	77.62	90.84	86.15	86.69	69.60	68.04	73.83	71.10	68.93
ReDet[20]	31.6M	80.10	88.81	82.48	60.83	80.82	78.34	86.06	88.31	90.87	**88.77**	87.03	68.65	66.90	79.26	79.71	74.67
DODet[6]	-	80.62	89.96	85.52	58.01	81.22	78.71	85.46	88.59	90.89	87.12	87.80	70.50	**71.54**	82.06	77.43	74.47
AOPG[5]	-	80.66	89.88	85.57	60.90	81.51	78.70	85.29	88.85	90.89	87.60	87.65	71.66	68.69	82.31	77.32	73.10
LSKNet[29]	31.0M	**81.64**	89.57	86.34	**63.13**	**83.67**	**82.20**	**86.10**	88.66	90.89	88.41	87.42	71.72	69.58	78.88	81.77	76.52
Single-stage																	
R^3Det[57]	41.9M	76.47	89.80	83.77	48.11	66.77	78.76	83.27	87.84	90.82	85.38	85.51	65.57	62.68	67.53	78.56	72.62
CFA[19]	-	76.67	89.08	83.20	54.37	66.87	81.23	80.96	87.17	90.21	84.32	86.09	52.34	69.94	75.52	80.76	67.96
SASM[23]	-	79.17	89.54	85.94	57.73	78.41	79.78	84.19	**89.25**	90.87	58.80	87.27	63.82	67.81	78.67	79.35	69.37
S^2Net[21]	-	79.42	88.89	83.60	57.74	81.95	79.94	83.19	89.11	90.78	84.87	87.81	70.30	68.25	78.30	77.01	69.58
R^3Det-GWD[58]	41.9M	80.23	89.66	84.99	59.26	82.19	78.97	84.83	87.70	90.21	86.54	86.85	**73.47**	67.77	76.92	79.22	74.92
RTMDet[35]	52.3M	80.54	88.36	84.96	57.33	80.46	80.58	84.88	88.08	**90.90**	86.32	87.57	69.29	70.61	78.63	80.97	**79.24**
R^3Det-KLD[59]	41.9M	80.63	89.92	85.13	59.19	81.33	78.82	84.38	87.50	89.80	87.33	87.00	72.57	71.35	77.12	79.34	78.68
Appr. MessDet	**15.3M**	80.36	88.45	85.50	59.10	81.51	79.97	84.49	88.32	90.89	87.39	87.20	69.55	69.63	78.16	**81.96**	73.23
Str. MessDet	18.1M	81.07	88.40	**86.54**	60.84	82.71	81.41	85.64	88.99	90.89	88.58	**88.05**	71.42	68.41	**83.78**	80.94	69.50

Table 7. **Comparison with state-of-the-art methods on the DOTA-v1.0 dataset** [15] with multi-scale training and testing. The mAP in parentheses refers to the COCO-style mAP.

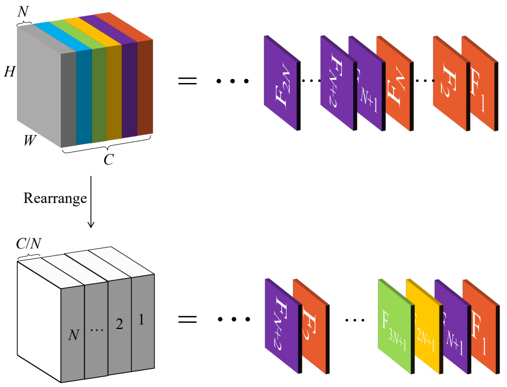


Figure 9. Illustration of Feature Rearrangement.

MessDet(without head)	Strictly	mAP	COCO-mAP
on DOTA-v1.0	✓	**78.51**	**51.96**
	✗	78.15	51.58
on DOTA-v1.5	✓	**72.42**	**46.02**
	✗	71.26	43.54

Table 8. Performance comparison of MessDet with two downsampling methods on DOTA-v1.0 and DOTA-v1.5.

Method	FLOPs(G)	FPS (img/s)	Training Time(H)
Str. MessDet	570	25.4	10.5
Appr. MessDet	378	38.2	7.7

Table 9. Information on Inference Speed, Training Time, and FLOPs.

In the head network of MessDet, features with inherent group properties from different orientations are fed to different branches. To achieve this, the features are rearranged along the channel dimension. The rearrangement process is shown in Fig. 9. For a rotation-equivariant feature, $\mathbf{X} \in \mathbb{R}^{C\times H\times W}$, where the feature map index of a certain channel is c, channels with the same remainder when dividing c by N are grouped together, thus grouping the channels generated by convolution kernels in different orientations.

9. Experiments Details and The Further Experiments

Our model is implemented using the MMYOLO [11] and MMRotate [64] frameworks and trained for 36 epochs on DOTA-v1.0, DOTA-v1.5 and DIOR-R. During training, we followed most mainstream methods [1, 20, 29, 51] by employing random rotation and random flipping to prevent over-fitting. The AdamW [34] optimizer is used with a base learning rate of 0.00025, weight decay of 0.05, and momentum of 0.9. The learning rate is gradually reduced to 1/20 of the base learning rate over the last half epochs using a cosine learning schedule. The experiments are conducted on 4 RTX 3090 GPUs with a batch size of 8. Following the exploration of ReDet [20] and FRED [26], this study set N, the number of orientation dimensions for the rotation-equivariant features in MessDet, to 8.

Here we adopt both single-scale and multi-scale training strategies. For single-scale training and testing on DOTA-v1.0 and DOTA-v1.5, we crop the original images into 1024×1024 patches with a stride of 824, yielding a pixel overlap of 200 between adjacent patches. For multi-scale training and testing, we first resize the original images to three scales (0.5, 1.0, and 1.5), and then crop them into 1024×1024 patches with a stride of 524, resulting in an overlap of 500 pixels.

Further Ablation Study on Rotation-Equivariant Downsampling. To further verify the performance gains brought by strictly equivariant downsampling, we conducted ablation experiments on the DOTA-v1.5 dataset using MessDet without the multi-branch head, and reported the COCO-style mAP, as shown in Tab. 8. It can be observed that, since the model performance on DOTA-v1.0 has reached a bottleneck in recent years, Str. MessDet shows only marginal improvement over its approximate counterpart. However, on the more challenging DOTA-v1.5 dataset, Str. MessDet achieves a significantly larger performance gain compared to Appr. MessDet.

Main Results with Multi-Scale Training on DOTA-v1.0. We also conducted multi-scale training experiments on the DOTA-v1.0 dataset for reference. Str. MessDet outperforms Appr. MessDet by 0.7 mAP, as shown in Tab. 7

Information on Inference Speed, Training Time, and FLOPs. We provide the relevant metrics in Tab. 9 for reference, based on DOTA-v1.0 with 4 RTX 3090 GPUs for single-scale training and a single RTX 3090 GPU for inference. The FLOPs are calculated based on an input image size of 1024×1024.