YOLO-DC: Enhancing object detection with deformable convolutions and contextual mechanism

Dengyong Zhang*, Chuanzhen Xu*, Jiaxin Chen*, Xin Liao†
* School of Computer and Communication Engineering, Changsha University of Science and Technology, Changsha 410114, China
E-mail: Dengyong Zhang, zhdy@csust.edu.cn; Chuanzhen Xu, chuanzhen@stu.csust.edu.cn; Jiaxin Chen, jxchen@csust.edu.cn
† College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China
E-mail: xinliao@hnu.edu.cn

***Abstract*—Existing object detection methods often optimize model structure, loss functions, and data preprocessing, but enhancements via convolutional techniques are often overlooked. Additionally, increasing network depth leads to the loss of key features. This paper introduces YOLO-DC, which enhances object detection with deformable convolution and contextual mechanisms. YOLO-DC includes a Deformable Convolutional Module (DCM) and a Contextual Fusion Downsampling Module (CFD). The DCM module uses a modified deformable convolution of Multiscale Spatial Channel Attention (MSCA) to extend the receptive field and improve feature extraction. The CFD integrates contextual and local features post-downsampling, reducing information loss. Compared to YOLOv8-N, YOLO-DC-N improves average precision (AP) by 3.5%, reaching 40.8% on the Microsoft COCO 2017 dataset, with similar FPS and inference time. The model demonstrates superior performance compared to other state-of-the-art detection algorithms. The source code is available at: https://github.com/Object-Detection-01/YOLO-DC.git.**

I. Introduction

Current object detectors rely on Convolutional Neural Networks (CNNs) and transformer-based models. Transformer models like DETR, DINO, and RT-DETR [1]–[3] captured long-term dependencies and achieved high accuracy but needed speed improvements compared to CNNs. The YOLO family, based on CNNs, balanced performance and efficiency. Since YOLOv1, enhancements like DarkNet [4], CSPNet [5], and ELAN [6] improved feature extraction. Enhancements like SPP [7], PANet [8], RepGFPN [9], and GD [10] enhanced multi-scale feature fusion. YOLOX [11] and YOLOv10 [12] improved performance by optimizing label allocation strategy. YOLO-MS improved accuracy with deep convolution [13].

Researchers proposed various convolution techniques to balance performance and efficiency, such as group convolution [14], dilated convolution [15], depthwise separable convolution [16], and deformable convolution [17]–[20]. Deformable convolution (DCNv1) adjusts the receptive field by learning offsets for each sample point [17]. Subsequent versions, DCNv2 [18], DCNv3 [19], and DCNv4 [20], introduced improvements like weight coefficient masking, weight sharing, and removal of softmax normalization. Additionally, attention and contextual mechanisms are crucial for improving model performance. Attention mechanisms such as CA [21], EMA [22], and EPSANet [23] encode semantic information at different scales, while GCNet [24], CGNet [25], and F-SSD [26] enhance performance by combining multiscale semantic information with contextual mechanisms.

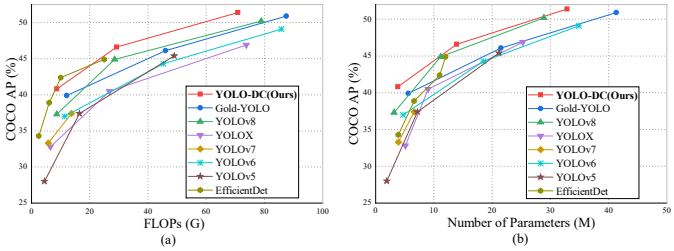


Fig. 1. Comparison with other SOTA object detection models on the COCO dataset: (a) AP vs. FLOPs; (b) AP vs. parameters. Notably, Faster R-CNN, RetinaNet, and DETR-DC models are not shown due to their high computational and parameter requirements. However, YOLO-DC outperforms these models, as detailed in Table I.

Despite significant progress by these techniques, the optimization space for convolutional enhancement was neglected in YOLOv8. In addition, increasing network depth led to information loss during feature extraction, causing biased gradient flow and affecting accuracy [27]. Consequently, YOLOv8 achieved sub-optimal performance. To address these issues, this paper proposes YOLO-DC, an object detection algorithm based on YOLOv8 [28]. As shown in Fig. 1, YOLO-DC achieves an optimal balance between computational load and parameter count, with superior performance compared to other state-of-the-art (SOTA) models. The main contributions of this paper include:

1) Improving DCNv2 by introducing a Multi-scale Spatial Channel Attention (MSCA), leading to the proposed DCN-MSCA convolution, which accurately extends the receptive field and constitutes a deformable convolutional module (DCM) for improved feature extraction.
2) Proposing the Context Fusion Downsampling (CFD) module to integrate contextual information and reduce redundancy after downsampling to improve information utilization.
3) Presenting YOLO-DC using DCM and CFD modules, which significantly outperforms existing SOTA models in comprehensive performance.

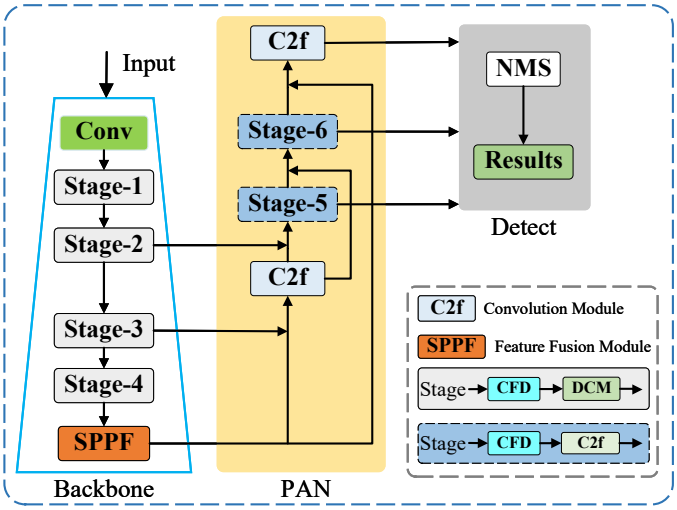


Fig. 2. YOLO-DC network structure diagram.

II. Proposed Method

A. Model Architecture

This paper presents YOLO-DC, an object detector based on YOLOv8, designed to enhance detection accuracy and address limitations in YOLOv8 convolution and information loss. The key components are the DCM and CFD modules. The CFD module replaces the downsampling operation in YOLOv8 [28], and the DCM module replaces the C2f module in the backbone. In each stage of the backbone network, the DCM provides robust feature extraction and an improved receptive field, while the CFD module reduces information loss during downsampling by enhancing features before feeding them into the DCM. In the PAN structure, each stage is fed into the C2f module after passing through the CFD module. This integrated approach significantly improves detection accuracy. The YOLO-DC network structure is shown in Fig. 2.

B. Contextual Fusion Downsampling Module (CFD)

The CFD module plays a crucial role in downsampling and information integration. It combines local characteristics with contextual semantic information and refines them with global contextual information, enhancing utilization efficiency. Positioned before the DCM module in the backbone and the C2f module in the neck, the CFD spans the entire network from spatial to semantic levels, facilitating rapid and accurate feature transfer and preventing information loss.

The CFD module first performs a 3×3 convolution with a stride of 2 for downsampling, followed by a regular convolution to collect local features and a dilated convolution to capture contextual features. It then passes through a joint feature extractor, including a concatenation layer, batch normalization, and SiLU activation to fuse local and contextual features. Finally, a global feature extractor, consisting of a global pooling layer and two fully connected layers, extracts features and generates a weight vector to guide joint feature fusion and produce the final output (See Fig. 3). Inspired by the Context Guided Block (CG Block) in CGNet [25], the CFD module combines surrounding contextual information with local features. While CG blocks are suitable for lightweight models, they may lack pronounced feature extraction capability. To maximize contextual information, the CFD module incorporates downsampling operations before the main convolutional module, facilitating comprehensive contextual information integration. This fused information enhances feature extraction and improves overall information extraction capability without significantly increasing computational load.

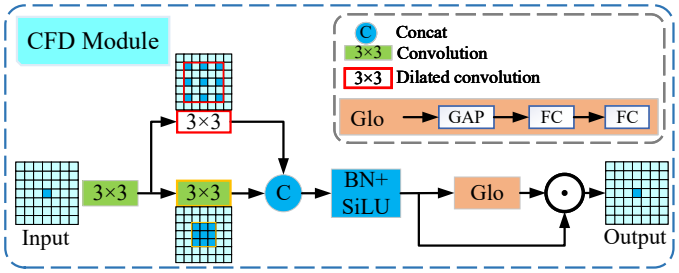


Fig. 3. CFD module schematic.

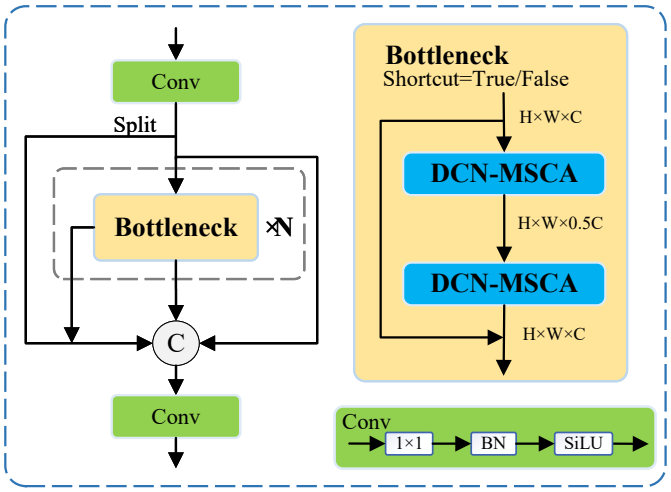


Fig. 4. DCM module schematic.

C. Deformable Convolution Module (DCM)

The DCM module integrates principles from the C2f module [28], preserving gradient flow. It combines residual connections with the CSPNet [5] structure to enhance learning. The DCM performs a convolution followed by a split into two branches. One branch passes through multiple Bottleneck modules with residual connections, then concatenates and fuses with the other branch and the output residuals of the Bottleneck module. The Bottleneck module in DCM employs DCN-MSCA to reduce irrelevant regions and improve feature extraction. The shortcut in the Bottleneck module connects at the backbone and disconnects at the neck (see Fig. 4).

Initially, we constructed the DCM with DCNv2, which slightly improved performance over the baseline model using normal convolution. However, the single convolution process in DCNv2 led to irrelevant regions. To address this, we added a Multiscale Spatial Channel Attention (MSCA) mechanism to DCNv2, enhancing offset attention in the X and Y directions. The improved DCN-MSCA method reconfigures offset generation in DCNv2, resulting in better accuracy and fewer

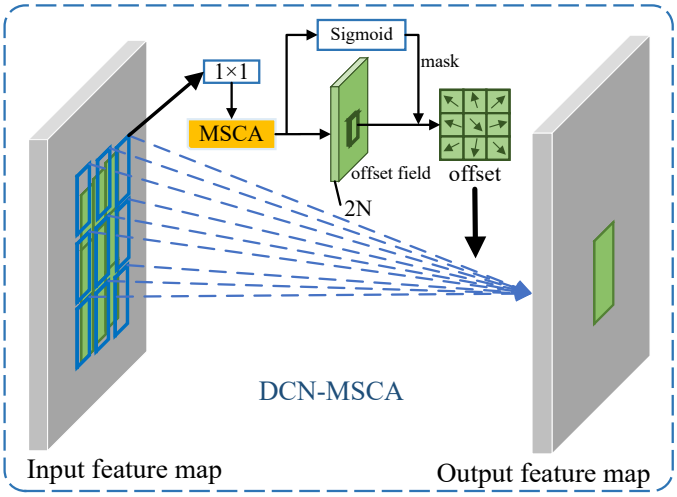


Fig. 5. DCN-MSCA convolution schematic.

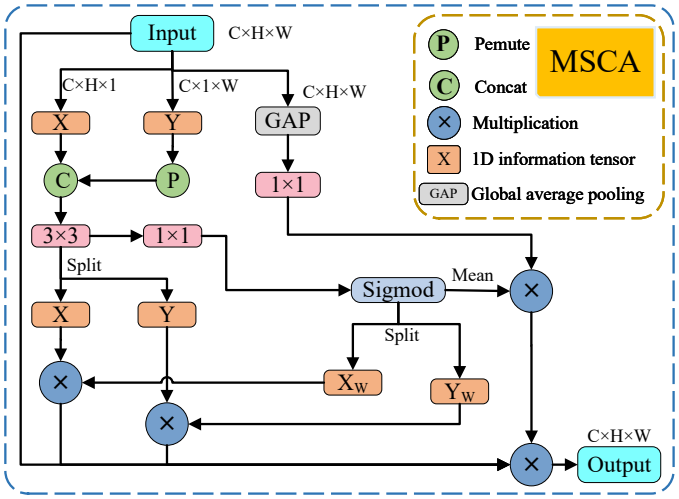


Fig. 6. MSCA mechanism schematic.

irrelevant regions (see Fig. 5).

The MSCA mechanism extends Coordinate Attention (CA) to tackle information loss by incorporating multiscale fusion. While CA decomposes the global pool into one-dimensional feature encoding for two spatial directions, its direct decomposition using two global pooling operations is somewhat crude. The subsequent splicing-then-convolution operation inadequately facilitates information exchange, contributing to suboptimal performance [21]. Our MSCA mechanism addresses this by integrating multiscale information to enhance accuracy by analyzing both spatial and channel information.

MSCA integrates multiscale information via CA, extracting weights from intermediate processes for bidirectional fusion. This enhances information interaction, with global information guiding both directions to mitigate loss. MSCA consists of three branches, each handling multiscale information at different gradients. The first branch transforms global pooling into one-dimensional codes for X and Y directions. The second branch convolves the first, extracting X_{weight} and Y_{weight} after Sigmoid activation to refine fusion accuracy. The third branch conducts global pooling on initial inputs, guiding encoded information from the second branch. The integrated output combines information from all three branches (see Fig. 6).

III. Experiment Evaluation

A. Experiment Setups

1) Datasets: We validated the performance of YOLO-DC using the Microsoft COCO 2017 dataset, the RUOD underwater object detection dataset, and the PASCAL VOC dataset (07+12). The COCO 2017 dataset includes 80 categories with 118,287 training images and 5,000 test images. The PASCAL VOC dataset (07+12) combines VOC 2007 and VOC 2012 training and validation sets, totaling 16,551 training images and 4,952 test images. The RUOD dataset, designed for underwater detection, comprises 9,800 training images and 4,200 test images.

2) Implementation Details: YOLOv8 was used as the baseline model. YOLO-DC was trained for 500 epochs, with mosaic enhancement disabled in the last 10 epochs to improve accuracy. The SGD optimizer with baseline hyperparameters was employed, using BCE loss for classification and DFL loss with CIoU loss for box regression. To ensure accurate evaluation, training was conducted from scratch without pre-training weights. Single-scale images (640×640) were used as input, following COCO evaluation metrics, reporting normalized AP at different IoU thresholds and AP for small, medium, and large targets (AP_s, AP_m, AP_l). Experiments were conducted on 2 NVIDIA RTX 3090 GPUs using PyTorch 2.0.

B. Comparisons

To evaluate YOLO-DC, we compared it against several SOTA models on the COCO 2017 dataset, including EfficientDet [31], RetinaNet [32], DETR-DC [1], Faster R-CNN [33], YOLOv7 [6], YOLOvX [11], Gold-YOLO [10], YOLO-MS [13], and the baseline YOLOv8 [28].

The experimental results in Table I showed that each version of YOLO-DC outperformed other models on the COCO 2017 dataset at the same computational overhead, demonstrating significant improvements in overall performance. The enhanced information utilization of the CFD module and the powerful feature extraction capability of the DCM module significantly improved the performance of YOLO-DC. Compared to YOLOv8-N, the AP of YOLO-DC-N increased by 3.5% to 40.8%, while maintaining similar computation and parameter counts, with almost the same inference time. The AP of YOLO-DC increased by 1.7% and 0.2% compared to YOLOv8-S and YOLOv8-M, respectively. Additionally, all versions of YOLO-DC outperformed EfficientDet, RetinaNet, and Faster R-CNN. Compared to YOLOv7-Tiny, YOLOX-N, and Gold-YOLO-N, YOLO-DC achieved AP improvements of 3.4%, 8%, and 0.9%, respectively. Both the S and M versions of YOLO-DC showed significant AP improvements over corresponding versions of these SOTA models. Compared with DETR-DC using ResNet-50 and ResNet-101, YOLO-DC-L outperformed them by 6.9% and 5.1% AP, respectively, with much lower computational overhead and inference time. Only YOLO-MS models had higher APs than YOLO-DC among the compared models. However, each version of YOLO-DC was

TABLE I
COMPARISON OF YOLO-DC WITH OTHER SOTA MODELS ON THE COCO 2017 DATASET. INFERENCE TIME ARE MEASURED WITH A BATCH SIZE OF 32.

Method	Input Size	$AP^{val}(\%)$	$AP^{val}_{50}(\%)$	$AP_s(\%)$	$AP_m(\%)$	$AP_l(\%)$	FPS	Latency	Params	FLOPs
EfficientDet-D0	512	34.3	54.2	12.0	37.3	50.2	95	10.5 ms	3.9 M	2.5 G
EfficientDet-D1	640	38.9	57.6	16.9	43.3	55.0	69	14.5 ms	6.6 M	6.1 G
EfficientDet-D2	768	42.4	61.3	20.5	46.0	57.4	51	19.6 ms	8.1 M	11 G
EfficientDet-D3	896	44.9	64.0	25.6	48.4	58.8	32	31.5 ms	12 M	25 G
RetinaNet-50	800	35.7	55.0	18.9	38.9	46.3	14	72.6 ms	29 M	165 G
RetinaNet-101	800	37.8	57.3	20.2	41.1	49.2	10	105.5 ms	38 M	205 G
Faster R-CNN	800	27.2	48.4	6.6	28.6	45.0	8	120 ms	42 M	180 G
Faster R-CNN +++	800	34.9	55.7	15.6	38.7	50.9	2	460 ms	60 M	246 G
DETR-DC5-R50	800	43.3	63.1	22.5	47.3	61.1	12	82.4 ms	41 M	187 G
DETR-DC5-R101	800	44.9	64.7	23.7	49.5	62.3	10	96.7 ms	60 M	253 G
YOLOv7-Tiny	416	33.3	49.9	-	-	-	1196	**0.8 ms**	6.2 M	5.8 G
YOLOv7-Tiny	640	37.4	55.2	19.9	41.1	50.8	519	1.9 ms	6.2 M	13.7 G
YOLOvX-N	416	32.8	50.3	14.0	35.5	48.3	1143	**0.9 ms**	5.1 M	6.5 G
YOLOvX-S	640	40.5	59.3	23.9	45.2	53.8	396	2.5 ms	9.0 M	26.8 G
YOLOvX-M	640	46.9	65.6	29.0	51.2	60.9	179	5.6 ms	25.3 M	73.8 G
Gold-YOLO-N	640	39.9	55.9	19.7	44.1	57.0	1030	1.0 ms	5.6 M	12.1 G
Gold-YOLO-S	640	46.1	63.3	25.3	50.2	62.6	446	2.2 ms	21.5 M	46.0 G
Gold-YOLO-M	640	**50.9**	68.2	32.3	55.3	66.3	220	4.5 ms	41.3 M	87.5 G
YOLO-MS-XS	640	**43.4**	**60.4**	23.7	48.3	60.3	131	7.6 ms	4.5 M	8.7 G
YOLO-MS-S	640	46.2	**63.7**	26.9	50.5	63.0	110	9.0 ms	8.1 M	15.6 G
YOLO-MS	640	**51.0**	**68.6**	33.1	56.1	66.5	80	12.3 ms	22.2 M	40.1 G
YOLOv8-N	640	37.3	52.6	15.3	35.6	54.7	734	1.4 ms	3.2 M	8.7 G
YOLOv8-S	640	44.9	60.8	23.6	47.1	65.7	387	2.6 ms	11.2 M	28.6 G
YOLOv8-M	640	50.2	67.2	28.9	53.6	69.6	176	5.7 ms	28.9 M	78.9 G
YOLO-DC-N (Ours)	640	**40.8**	56.9	16.6	39.6	60.0	676	1.5 ms	3.9 M	8.9 G
YOLO-DC-S (Ours)	640	**46.6**	63.5	24.6	48.5	65.8	334	3.0 ms	13.9 M	29.2 G
YOLO-DC-M (Ours)	640	**50.4**	67.3	27.9	53.9	69.7	147	6.8 ms	32.9 M	70.9 G

TABLE II
COMPARATIVE EVALUATION OF YOLO-DC WITH OTHER SOTA MODELS ON THE PASCAL VOC (07+12) AND RUOD DATASETS.

Mothod	Params	FLOPs	PASCAL VOC(07+12)		RUOD	
			$AP^{val}(\%)$	$AP^{val}_{50}(\%)$	$AP^{val}(\%)$	$AP^{val}_{50}(\%)$
YOLOv5-N [29]	1.9 M	4.5 G	45.1	72.5	53.6	72.1
YOLOv5-S	7.2 M	16.5 G	53.0	76.2	58.8	79.4
YOLOv6-N [30]	4.7 M	11.4 G	60.5	82.0	59.7	84.2
YOLOv7-Tiny [6]	6.2 M	13.7 G	53.8	79.3	57.2	85.3
YOLOv8-N [28]	3.2 M	8.7 G	59.1	79.9	61.9	85.3
YOLO-DC-N (Ours)	3.9 M	8.9 G	**62.6**	**82.4**	**62.8**	**85.6**

faster; for example, the inference time for YOLO-DC-N was only 20% of that for YOLO-MS-XS.

For a comprehensive evaluation, we conducted comparative experiments on the PASCAL VOC dataset (07+12) and the RUOD dataset, including models such as YOLOv5 (N, S), YOLOv6-N, YOLOv7-Tiny, and YOLOv8-N.

Table II presents the comparison results on the PASCAL VOC (07+12) and RUOD datasets. YOLO-DC-N achieved a significantly higher AP of 62.6% on PASCAL VOC, an improvement of 3.5% over YOLOv8-N, outperforming all selected SOTA models. On the RUOD dataset, YOLO-DC-N achieved an AP of 62.8%, a 1.1% increase over YOLOv8-N. Fig. 7 shows detection results comparison between YOLO-DC and YOLOv8, highlighting the superior performance of YOLO-DC in challenging environments.

Experiments show that YOLO-DC outperforms several SOTA algorithms across multiple datasets in parameters, computation, inference time, and detection accuracy.

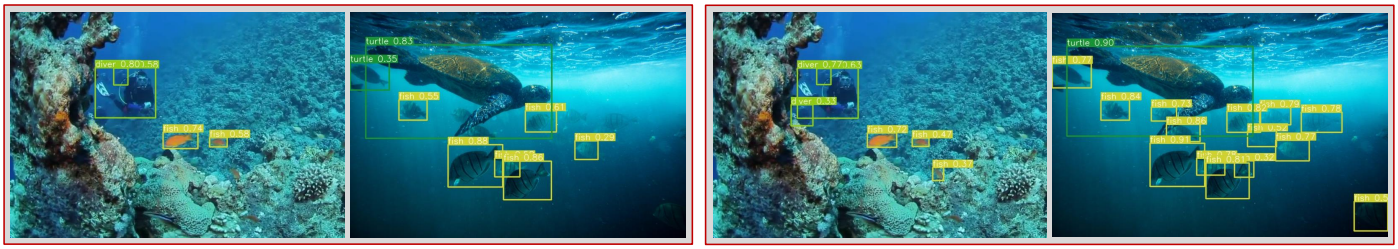
C. Ablation Study

To verify the effectiveness of our proposed improvements, we conducted ablation studies on DCNv2, DCM, CFD, and MSCA, using YOLOv8-N as the baseline model, on the Microsoft COCO 2017 dataset (Table III).

Starting with YOLOv8-N, which had an AP of 37.3%, replacing normal convolution in the C2f module with DCNv2 increased AP by 1.33% to 38.63%. Further enhancing DCNv2 with MSCA led to an additional 0.4% increase, reaching 39.03%. The CFD module alone improved AP by 1.14% to 38.44%. Combining DCM and CFD in YOLO-DC resulted in a significant AP improvement of 3.5%, reaching 40.8%. Despite a slight increase in parameters and computational load, inference time remained almost unchanged, indicating a favorable performance tradeoff. Fig. 8 illustrates the effectiveness of feature extraction for ordinary convolution, DCNv2, and DCN-MSCA in the model, highlighting the superior feature extraction capability of DCN-MSCA after applying MSCA.

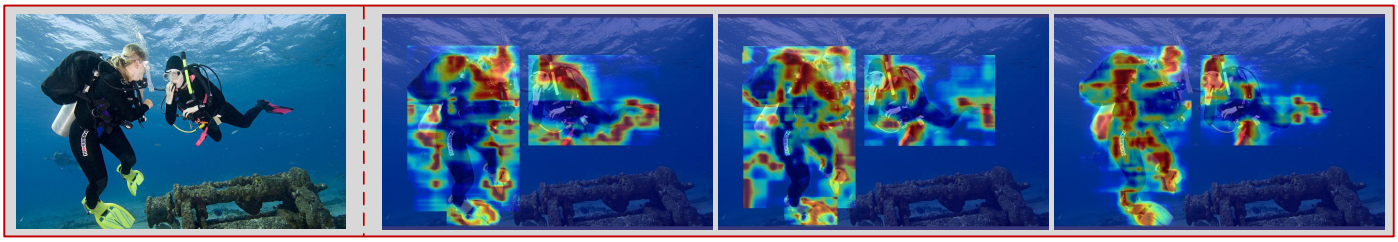
TABLE III
ABLATION STUDY ON DIFFERENT MODULES USING YOLOV8-N AS THE BASELINE MODEL: EVALUATING PERFORMANCE ON THE COCO 2017 DATASET.

Method	$AP^{val}(\%)$	$AP^{val}_{50}(\%)$	$AP_s(\%)$	$AP_m(\%)$	$AP_l(\%)$	FPS	Latency	Params	FLOPs
YOLOv8-N(Baseline)	37.30	52.61	15.28	35.59	54.66	734	1.4 ms	3.2 M	8.7 G
+DCNv2	38.63	54.44	15.62	37.63	55.91	756	1.3 ms	3.2 M	7.4 G
+DCN-MSCA(DCM)	39.03	54.97	15.72	37.93	56.32	724	1.4 ms	3.2 M	7.9 G
+CFD	38.44	53.83	15.24	36.81	55.68	789	1.3 ms	3.8 M	9.8 G
YOLO-DC(DCM+CFD)	**40.80**	**56.89**	**16.62**	**39.61**	**60.04**	676	1.5 ms	3.9 M	8.9 G



(a) YOLOv8 (b) YOLO-DC

Fig. 7. Comparison of image detection results selected from the RUOD dataset. (a) is the detection result of YOLOv8. (b) is the detection result of YOLO-DC.



(a) Input image (b) Ordinary convolution (c) DCNv2 (d) DCN-MSCA

Fig. 8. Comparison plots of feature extraction results with different convolutions using YOLOv8 as the base model. (a) shows the input image, while (b) to (d) display the visualized heatmaps (GradCAM) extracted by the model at the convolution stage of the backbone network after applying normal convolution, DCNv2, and DCN-MSCA, respectively.

IV. CONCLUSION

This paper introduces YOLO-DC, an object detection model with deformable convolution. YOLO-DC features a Deformable Convolution Module (DCM) and a Context Fusion Module (CFD). The DCM enhances feature extraction with multi-scale spatial channel attention, while the CFD reduces information loss after downsampling. Experiments show that YOLO-DC outperforms other state-of-the-art models in both general and underwater object detection.

REFERENCES

[1] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European Conference on Computer Vision (ECCV)*, 2020, pp. 213–229.

[2] H. Zhang, F. Li, S. Liu, *et al.*, "Dino: Detr with improved denoising anchor boxes for end-to-end object detection," in *The Eleventh International Conference on Learning Representations (ICLR)*, 2022.

[3] Y. Zhao, W. Lv, S. Xu, *et al.*, "Detrs beat yolos on real-time object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 16 965–16 974.

[4] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[5] C.-Y. Wang, H.-Y. M. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I.-H. Yeh, "Cspnet: A new backbone that can enhance learning capability of cnn," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020.

[6] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 7464–7475.

[7] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Scaled-yolov4: Scaling cross stage partial network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 13 024–13 033.

[8] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 8759–8768.

[9] X. Xu, Y. Jiang, W. Chen, Y. Huang, Y. Zhang, and X. Sun, "Damo-yolo: A report on real-time object detection design," *arXiv preprint arXiv:2211.15444*, 2022.

[10] C. Wang, W. He, Y. Nie, *et al.*, "Gold-yolo: Efficient object detector via gather-and-distribute mechanism," in *Neural Information Processing Systems (NIPS)*, vol. 36, 2023, pp. 51 094–51 112.

[11] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: Exceeding yolo series in 2021," *arXiv preprint arXiv:2107.08430*, 2021.

[12] A. Wang, H. Chen, L. Liu, *et al.*, "Yolov10: Real-time end-to-end object detection," *arXiv preprint arXiv:2405.14458*, 2024.

[13] Y. Chen, X. Yuan, R. Wu, J. Wang, Q. Hou, and M.-M. Cheng, "Yolo-ms: Rethinking multi-scale representation learning for real-time object detection," *IEEE Transactions on Cognitive and Developmental Systems (TCDS)*, 2023.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems (NIPS)*, vol. 25, 2012.

[15] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *The Eleventh International Conference on Learning Representations (ICLR)*, 2016.

[16] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1251–1258.

[17] J. Dai, H. Qi, Y. Xiong, *et al.*, "Deformable convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 764–773.

[18] X. Zhu, H. Hu, S. Lin, and J. Dai, "Deformable convnets v2: More deformable, better results," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 9300–9308.

[19] W. Wang, J. Dai, Z. Chen, *et al.*, "Internimage: Exploring large-scale vision foundation models with deformable convolutions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 14 408–14 419.

[20] Y. Xiong, Z. Li, Y. Chen, *et al.*, "Efficient deformable convnets: Rethinking dynamic and sparse operator for vision applications," *arXiv preprint arXiv:2401.06197*, 2024.

[21] Q. Hou, D. Zhou, and J. Feng, "Coordinate attention for efficient mobile network design," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 13 713–13 722.

[22] D. Ouyang, S. He, G. Zhang, *et al.*, "Efficient multi-scale attention module with cross-spatial learning," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5.

[23] H. Zhang, K. Zu, J. Lu, Y. Zou, and D. Meng, "Epsanet: An efficient pyramid squeeze attention block on convolutional neural network," in *Asian Conference on Computer Vision (ACCV)*, 2023, pp. 541–557.

[24] Y. Cao, J. Xu, S. Lin, F. Wei, and H. Hu, "Gcnet: Non-local networks meet squeeze-excitation networks and beyond," *International Conference on Computer Vision Workshop (ICCVW)*, pp. 1971–1980, 2019.

[25] T. Wu, S. Tang, R. Zhang, J. Cao, and Y. Zhang, "Cgnet: A light-weight context guided network for semantic segmentation," *IEEE Transactions on Image Processing (TIP)*, vol. 30, pp. 1169–1179, 2020.

[26] J.-S. Lim, M. Astrid, H.-J. Yoon, and S.-I. Lee, "Small object detection using context and attention," in *International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, IEEE, 2021, pp. 181–186.

[27] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, *Yolov9: Learning what you want to learn using programmable gradient information*, 2024. arXiv: 2402.13616.

[28] G. Jocher, A. Chaurasia, and J. Qiu, *Ultralytics YOLOv8*, https : / / github . com / ultralytics / ultralytics, version 8.0.0, Jan. 2023.

[29] G. Jocher, *YOLOv5 by Ultralytics*, https://github.com/ultralytics/yolov5, version 7.0, May 2020.

[30] C. Li, L. Li, H. Jiang, *et al.*, "Yolov6: A single-stage object detection framework for industrial applications," *arXiv preprint arXiv:2209.02976*, 2022.

[31] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 10 781–10 790.

[32] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2980–2988.

[33] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 39, pp. 1137–1149, 2016.