Information Fusion 129 (2026) 104041

Contents lists available at ScienceDirect

Information Fusion

journal homepage: www.elsevier.com/locate/inffus

A real-time surface defect detection model based on adaptive feature information selection and fusion

Li-Juan Liu [a], Shao-Qi Sun [b], Hamid Reza Karimi [c,*]

[a] School of Information Science, Beijing Language and Culture University, Beijing, 100083, China
[b] School of Railway Intelligent Engineering, Dalian Jiaotong University, No. 794 Huanghe Road, DaLian, 116000, China
[c] Department of Mechanical Engineering, Politecnico di Milano, 20156, Italy

ARTICLE INFO

Keywords:
Deep learning
Object detection
Feature fusion pyramid network
Steel surface defect detection

ABSTRACT

In contemporary computer vision, You Only Look Once (YOLO) has become a benchmark for object detection, widely used in domains from intelligent manufacturing-such as industrial quality control and automated inspection-to real-time video surveillance. For example, detecting surface defects on steel products or electronic components in production lines relies on such algorithms to maintain high quality and safety. Despite YOLO's excellent speed and accuracy in many tasks, it still faces difficulties in certain challenging conditions, notably high dynamic range scenes, complex backgrounds, and the detection of small or subtle objects. These conditions are common in practice-for instance, on shiny metal surfaces with uneven lighting or in busy surveillance scenes-where conventional YOLO models struggle to capture fine details reliably. To overcome these limitations, we propose an improved YOLO-based framework featuring a novel Dynamic Cross-Scale Feature Fusion Module (Dy-CCFM) and a Dual-path Downsampling Convolution Module (DDConv). These modules enhance multi-scale feature representation and preserve detail under extreme lighting and background clutter, which is crucial for monitoring in complex environments. Additionally, we employ the Minimum Point Distance Intersection over Union (MPDIoU) as an optimized loss function for bounding box regression, significantly improving the localization of small objects. Thanks to these innovations, the model achieves a mean Average Precision (mAP) of 75.1 % on the challenging Northeastern University surface defect (NEU-DET) dataset, while the smallest variant is only 1.6M in size. Compared to YOLOv8, our approach improves mAP by 2.1 % while also delivering higher inference speed (FPS), and it surpasses the Detection Transformer (DETR) by 5.0 % mAP. The model further demonstrates excellent generalization on the Google Cloud 10 Defect Detection (GC10-DET) dataset. This enhanced detection algorithm not only improves performance but also offers significant practical value in intelligent manufacturing and automated inspection systems, intelligent video surveillance, and autonomous vehicles, where reliable real-time detection of small defects or targets is critical.

1. Introduction

Deep learning and computer vision technologies have significantly enhanced object detection capabilities, driving their widespread application across various fields [1]. Among them, surface defect detection for steel has emerged as a critical research focus in intelligent manufacturing, playing a pivotal role in ensuring product quality and production safety [2]. As a fundamental material in industrial infrastructure and high-end equipment manufacturing, steel is highly susceptible to surface defects, which not only compromise product appearance but may also induce structural fatigue and fractures, leading to substantial economic losses and safety risks. Similarly, in electronic device manufacturing, micro-defects such as solder joint oxidation, micro-cracks, or scratches can impair conductivity and system stability, potentially resulting in complete system failure. Therefore, defect detection technology serves not only as a core quality assurance mechanism in heavy industry and precision manufacturing but also as a key support for safe operation and production cost control.

The introduction provides a flowchart summarizing the methodology and contributions to guide readers and enhance engagement [3].

In actual production environments, steel surfaces inevitably exhibit various types of defects such as cracks, scratches, and dents. Traditional inspection methods heavily rely on manual visual inspection, which is inefficient, labor-intensive, and highly dependent on the operator's experience and condition. Fatigue and subjective judgment may easily result in missed or false detections. Moreover, manual inspection is

* Corresponding author.
E-mail address: hamidreza.karimi@polimi.it (H.R. Karimi).

https://doi.org/10.1016/j.inffus.2025.104041
Received 30 June 2025; Received in revised form 2 December 2025; Accepted 3 December 2025
Available online 11 December 2025

inadequate for high-speed production lines or the identification of extremely small defects, thereby limiting the progress of industrial intelligence.

Against this backdrop, traditional computer vision methods were initially adopted for automated defect detection. These methods typically depend on handcrafted features such as edge detection, grayscale distribution, and texture modeling, in conjunction with thresholding or rule-based classifiers for recognition tasks. While these methods can achieve basic detection under certain conditions, they generally suffer from weak generalization ability, strong parameter dependency, and difficulty adapting to complex backgrounds and variable defect patterns. For instance, under interference conditions such as oil stains, high reflectivity, or overlapping scratches, traditional algorithms lack robustness and the ability to continuously improve through large-scale data learning.

With the advancement of deep learning, end-to-end object detection methods have gradually become mainstream. Object detection, a core task in computer vision, aims to identify and localize targets in images or videos, and its development is largely driven by breakthroughs in convolutional neural networks (CNNs) [4]. The current mainstream detection algorithms fall into two categories: one-stage methods, such as SSD [5], DETR [6], and the YOLO series [7], which directly output class and location predictions and are suitable for real-time applications; and two-stage methods, such as the R-CNN family [8], which first generate region proposals and then perform classification, offering higher accuracy at the expense of greater computational complexity [9]. Among them, one-stage methods have seen wide adoption in infrared small target detection and industrial defect inspection due to their fast inference and low deployment cost [10].

In the field of steel surface defect detection, numerous YOLO-based studies have been proposed. For example, Kou et al. [11] developed a defect detection model based on YOLOv3 for steel strips, achieving improvements in both accuracy and speed. Other studies introduced channel reorganization mechanisms to accelerate inference and enhance precision [12], or integrated multiple detection architectures to improve detection performance for small and irregular defects [13]. YOLOv5 to YOLOv7 have shown outstanding performance in practical industrial applications: YOLOv5 features a lightweight architecture and optimized loss functions, demonstrating robust capabilities under complex backgrounds and for small targets [15]; YOLOv6 incorporates EfficientNet and Focal Loss, further enhancing efficiency and adaptability [16]; YOLOv7 strengthens deep feature extraction and feature fusion mechanisms [17].

With the continuous evolution of deep learning, higher-performance models in the YOLO family have been developed. YOLOv8 introduces anchor-free detection heads, simplifying the prediction process [18]; YOLOv9 focuses on network structure simplification and optimization of feature extraction modules [19]; YOLOv10 integrates attention mechanisms to dynamically select features based on object size, thereby enhancing generalization [20].

Despite the significant progress in detection speed and accuracy, the YOLO series still faces challenges in surface defect detection for steel, particularly in dealing with multi-scale targets, complex background interference, and the recognition of tiny defects. Key limitations include the insufficient representation capacity of shallow features for small targets, limited multi-scale feature fusion capabilities, and inadequate robustness against highly dynamic changes and complex textured backgrounds [21–23].

To address these issues and further enhance the applicability of YOLO in industrial defect detection, this paper proposes an improved model-Dynamic-YOLO-and introduces three key modules to achieve superior detection performance:

DDConv (Dual-path Downsampling Convolution): Unlike traditional downsampling methods (e.g., max pooling or single-path convolution), DDConv adopts a dual-path structure to simultaneously extract local detail and global context, thereby improving fine-grained defect responsiveness while maintaining a lightweight design and high inference efficiency.

Dy-CCFM (Dynamic Cross-Scale Feature Fusion Module): In contrast to static feature fusion structures (e.g., FPN, PAN), Dy-CCFM incorporates a dynamic weighting mechanism to adaptively fuse multi-scale semantic and spatial information, significantly improving detection stability in complex backgrounds and for small targets.

MPDIoU (Minimum Point Distance IoU) Loss Function: By incorporating minimum point distance information, MPDIoU effectively addresses the regression issue where the predicted box and ground truth have similar shapes but different sizes, thereby enhancing the localization accuracy of small defects such as cracks and scratches [24].

Based on these enhancements, the proposed Dynamic-YOLO model achieves a mean average precision (mAP) of 75.1 % on the NEU-DET dataset, 2.1 % higher than YOLOv8 and 5.0 % higher than DETR. Moreover, the model contains only 1.6M parameters, demonstrating excellent inference efficiency and deployment adaptability. The lightweight design and high robustness of this model make it suitable not only for industrial defect detection but also for cross-domain applications. Especially in complex industrial backgrounds, such as steel surface defect detection, the model shows significant performance improvement.

Notably, video surveillance, intelligent transportation, and public safety domains face similar challenges, such as varying backgrounds and complex target shapes. In recent years, deep transfer learning (DTL) and deep domain adaptation (DDA) methods have been widely applied to object detection in video surveillance, greatly enhancing cross-domain generalization [25]. Based on this theoretical background, the Dynamic-YOLO model also demonstrates potential as a benchmark model for video detection, particularly in challenging scenarios like low-light conditions, small targets, and dynamic backgrounds. Therefore, in addition to steel surface defect detection, this model can also serve as an efficient benchmark model for cross-domain tasks such as video surveillance, intelligent transportation, and public safety, adapting to the data requirements and visual environments of various domains.

2. Related work

2.1. Conv used for object detection

In recent years, researchers have analyzed standard convolution operations from multiple perspectives and proposed various novel convolutional operators to enhance network performance. Li et al. pointed out that the spatial sharing of convolution kernel parameters limits the network's ability to model diverse regional features and hinders the capture of long-range dependencies. To address this issue, they introduced the Involution operator, which reverses the conventional feature transformation process of convolution to improve network flexibility [26]. Qi et al. proposed Distribution Shifted Convolution (DscConv) based on deformable convolution. However, the free offset mechanism of deformable convolution tends to lose fine-grained structural information, prompting the introduction of DSConv to enhance the detection of slender structural targets [27]. Zhang et al. explored spatial attention mechanisms, which to some extent alleviate the parameter sharing problem in convolution [28]. Nonetheless, common spatial attention mechanisms such as Convolutional Block Attention Module (CBAM) [29] and Coordinate Attention (CA) [30], while beneficial for performance, still fall short in maintaining stable spatial representations under large-scale convolutions or in highly disturbed environments.

Furthermore, the study by Islam et al. [31] revealed that convolutional models often struggle to maintain stable spatial representations under external disturbances such as seasonal changes. By introducing convolutional auto-encoders and independent component analysis (ICA), the authors constructed a robust spatial representation system that significantly improved robot localization stability in dynamic environments. This finding further highlights the limitations of existing attention-based convolutional mechanisms in modeling structural

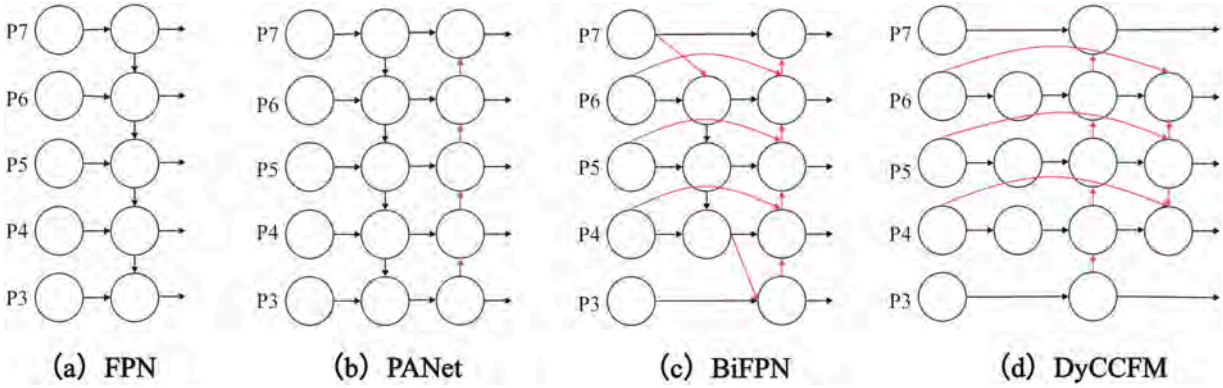


Fig. 1. Network structure diagrams of FPN, PANet, BiFPN, and Dy-CCFM.

invariance under real-world conditions, and supports the need for more efficient and robust spatial modeling architectures.

Although the aforementioned techniques have led to performance improvements, they have not fundamentally solved the problems of parameter efficiency and spatial representation capacity in convolution. In contrast, the proposed DDConv module achieves efficient feature extraction with extremely low computational cost and is well suited for detection tasks that require both fine structural detail and global context modeling.

2.2. Multi-scale features used for object detection

In the field of object detection, feature maps at different scales contain critical positional information for objects of various sizes. Large-scale feature maps can reveal the details and locations of small objects, while small-scale feature maps capture complex information and localization for larger objects. The Feature Pyramid Network (FPN) leverages the complementary nature of multi-scale information, providing an efficient architecture for multi-scale feature fusion through cross-scale connections and information flow, thereby improving detection accuracy [31]. However, FPN faces limitations in handling high-level target information. To address this issue, methods such as PANet [32], BiFPN [33], and Balanced FPN have introduced multiple enhancements to FPN to strengthen feature fusion. CE-FPN utilizes attention mechanisms to achieve selective feature fusion, while FaPN [34] incorporates feature alignment modules to improve fusion accuracy and address potential misalignment issues across scales.

Although these multi-scale feature fusion methods provide valuable insights for steel surface defect detection, they are primarily designed based on the characteristics of natural images. While some methods show effectiveness in defect detection, they do not fully account for the specific characteristics of steel surface defect images, thereby limiting detection efficacy. To address these challenges, this study proposes the Dynamic Cross-Feature Fusion Module (Dy-CCFM), designed to tackle the multi-scale and complex background challenges in steel surface defect detection. Dy-CCFM employs an innovative dynamic feature selection and fusion mechanism, effectively extracting and integrating features from various scales, thereby enhancing the model detection capability across defect scales. Dy-CCFM uses high-level features as weights to dynamically optimize the selection and fusion of low-level features, allowing high-level semantic information to better support small-scale defect detection. This dynamic selective fusion strategy enables precise information flow between features at different scales, enhancing the model ability to detect fine and marginal defects [35–37].

Furthermore, the Dy-CCFM architecture is optimized for complex industrial environments, balancing high detection accuracy with computational efficiency, and demonstrating superior performance in steel surface defect detection tasks, Fig. 1 illustrates the design of Dy-CCFM and compares it with other FPN networks.

3. Methods

In the framework, Fig. 2 shows our overall design, integrating the CCFM [17], DySample [40], and DDConv modules. The CCFM module realizes efficient feature fusion and transmission through the serial and parallel connections of different operational units. The DDConv module optimizes the reduction of feature dimensions and the adjustment of spatial resolution, enabling the model to handle features of various scales. The DySample module is responsible for dynamic sampling, adjusting the sampling strategy based on context information to effectively capture key features. We named it Dynamic-YOLO because of the presence of Dy-CCFM, which allows for feature selection operations. The model does not extract all features but selects the effective ones, filtering out the irrelevant features. Therefore, our model is more dynamic in feature selection. Next, we will introduce these key components in detail.

3.1. Dual-path downsampling convolution module

In steel surface defect detection, target regions are often small, have blurred edges, and vary widely in shape. They are frequently accompanied by specular reflections, overlapping scratches, and complex textures. Under such challenging conditions, traditional downsampling modules in convolutional networks show clear limitations.

On one hand, conventional methods such as max pooling or strided convolution lack awareness of spatial structure. As a result, they tend to discard fine local details-especially small edges and subtle texture patterns-that are crucial for detecting fine-grained defects. On the other hand, standard convolutions can preserve rich feature information but require high computational cost, making them unsuitable for real-time or lightweight industrial applications.

To overcome these problems, we propose a Dual-path Downsampling Convolution (DDConv) module that enhances both accuracy and efficiency. The module optimizes the input feature map to extract richer and more discriminative representations.

Let the input tensor x have the shape (b, c_1, w, h), where b is the batch size, c_1 is the number of input channels, and w and h are the spatial dimensions.

The DDConv module first performs average pooling and then splits the feature map into two parallel branches, X_1 and X_2, each with the shape $(b, c_1/2, w, h)$. These two branches use different processing strategies to balance information preservation and computational efficiency.

For the X_1 branch: The input tensor X_1 is divided into four non-overlapping sub-tensors, each with the shape $(b, c_1, w/2, h/2)$, by splitting the spatial dimensions:

$$X_{\text{top\_left}} = X[\dots, :: 2, :: 2] \tag{1}$$

$$X_{\text{top\_right}} = X[\dots, :: 2, 1 :: 2] \tag{2}$$

$$X_{\text{bot\_left}} = X[\dots, 1 :: 2, :: 2] \tag{3}$$

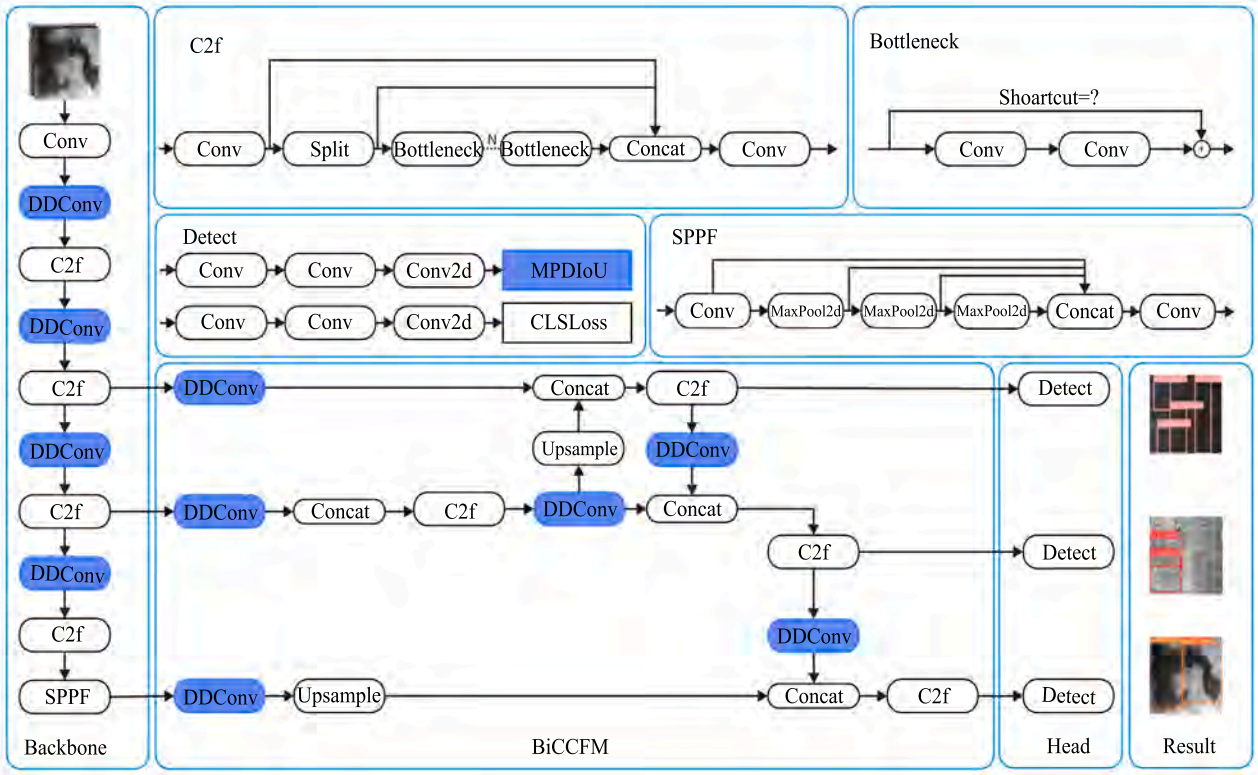


Fig. 2. Compared to YOLOv8, the parts with a darker background color indicate the modified modules.

$$X_{\text{bot\_right}} = X[\ldots, 1::2, 1::2] \tag{4}$$

These four sub-tensors are then concatenated along the channel dimension, resulting in a tensor with the shape $(b, 4c_1, w/2, h/2)$. A convolutional operation is then applied, producing an output tensor with shape $(b, \text{out\_channels}, w/2, h/2)$.

Parameter Analysis for X_1: For the convolution operation in the 'Focus' module, the input channels are $4c_1$ (due to concatenation), and the output channels are out_channels. Assuming the kernel size is $K \times K$, the number of parameters for this convolution is:

$$\text{Parameters}_{\text{Focus}} = 4\, c_1\, K^2\, \text{out\_channels} \tag{5}$$

For the X_2 branch: The X_2 branch applies max pooling followed by a convolution operation, resulting in an output tensor of the same shape $(b, \text{out\_channels}, w/2, h/2)$. If the convolution kernel size is 1×1, the number of parameters for this convolution is:

$$\text{Parameters}_{\text{cv2}} = \left(\frac{c_1}{2}\right) \times \frac{\text{out\_channels}}{2} \times 1^2 = \frac{c_1 \times \text{out\_channels}}{4} \tag{6}$$

Concatenation of Outputs: After processing by both branches, the output tensors are concatenated along the channel dimension, resulting in the final output tensor Y with shape $(b, c_2, w/2, h/2)$, where $c_2 = \text{out\_channels}$.

Total Number of Parameters for DDConv: The total number of parameters in the DDConv module is the sum of the parameters from both branches:

$$\text{Total Parameters}_{\text{DDConv}} = 4c_1 \times K^2 \times \frac{\text{out\_channels}}{2} + \frac{c_1 \times \text{out\_channels}}{4} \tag{7}$$

Simplifying this expression, we obtain:

$$\text{Total Parameters}_{\text{DDConv}} = c_1 \times \text{out\_channels} \times \frac{K^2}{4} + \frac{c_1 \times \text{out\_channels}}{4} \tag{8}$$

Parameter Optimization Analysis: In comparison to a standard convolution operation, where the parameter count is given by:

$$\text{Standard Parameters} = c_1 \times K^2 \times c_2 \tag{9}$$

DDConv significantly reduces the number of parameters. The reason for this reduction is twofold:

- The use of the 'Focus' module compresses the spatial dimensions by a factor of 2 (from $W \times H$ to $W/2 \times H/2$), thereby reducing the number of operations and parameters in the convolution.
- The use of 1×1 convolutions in the second branch further reduces the parameter count by limiting the channel interactions to just a linear transformation.

Thus, the total number of parameters in DDConv is approximately one-quarter of that in standard convolutions, primarily due to the reduced spatial dimensions and the smaller convolution kernel sizes. This makes DDConv an efficient alternative, offering reduced computational overhead while still capturing fine-grained feature representations.

Applications: The DDConv module is effective for tasks such as image classification and object detection. It helps the model detect small visual differences while keeping computational cost low. Additionally, DDConv can be easily integrated into modern object detection frameworks. The complete architecture of the module is shown in Fig. 3.

Remark 1. We propose the DDConv module to improve feature extraction while reducing computational complexity by splitting the input image into two branches. The main branch divides the image into four segments and extracts features from each, making it well-suited for detecting small objects in surface defect inspection. The auxiliary branch uses pooling followed by convolution, avoiding traditional 3x3 convolutional downsampling, which reduces computational load and parameter count while preserving rich features. Finally, features from both branches are fused, combining diverse information and completing the downsampling process, resulting in more efficient and representative

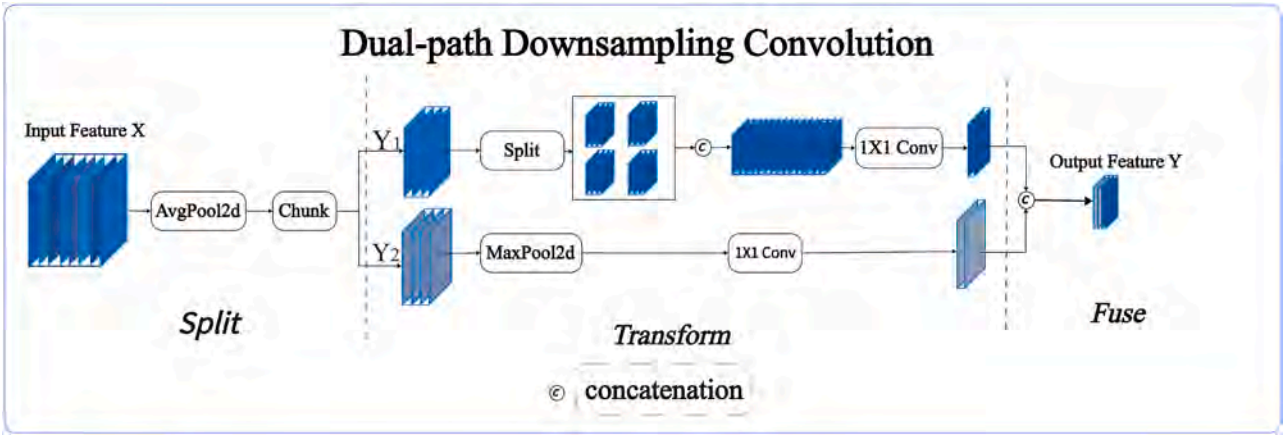


Fig. 3. Schematic Diagram of the Dual-Path Downsampling Convolution Module.

feature extraction. This design contributes to the high efficiency of our proposed Dynamic-YOLO.

3.2. Dynamic-cross scale feature fusion module

In steel surface defect detection, defects vary widely in type and scale, and the images often contain complex textures, uneven lighting, and significant background noise. These challenges make multi-scale feature fusion a critical factor in improving detection accuracy. Traditional fusion frameworks, such as FPN and PANet, provide basic multi-scale processing but rely on static fusion strategies. As a result, they cannot adapt to different target scales or varying feature distributions, which reduces performance on small, low-contrast, or blurred defects. Furthermore, their fixed architectures often struggle to balance detailed spatial information from shallow layers with high-level semantic cues from deeper layers, leading to redundant, less discriminative, or incomplete feature representations.

To address these limitations, we propose a Dynamic Cross-Scale Feature Fusion Module (Dy-CCFM) that enhances adaptability and robustness by dynamically adjusting the fusion strategy based on the characteristics of the input image. Dy-CCFM integrates information from multiple scales and employs multiple feature extraction paths to generate richer and more representative feature maps. By jointly modeling both spatial and channel-wise interactions, it improves feature utilization, allowing the network to more effectively capture subtle variations and complex patterns on the steel surface. This dynamic and context-aware fusion approach helps the model maintain high detection accuracy across a wide variety of defect types and challenging imaging conditions.

Analysis of the architecture shown in Fig. 1(c) revealed that adding extra connections from shallow to deep layers resulted in a 15 % increase in inference time, leading to the abandonment of this structure and its improvement by replacing traditional upsampling operations with DySample. With its adaptive upsampling capability, DySample better preserves scale and detail information, enabling the model to distinguish minor defects from normal surface textures. DySample aims to improve the performance of feature up-sampling by accurately modeling geometric information. The core idea is to use a dynamic sampling strategy to re-sample input feature maps, generating more refined and adaptable up-sampled feature maps. Specifically, DySample achieves its functionality through the following two key components:

(1) Dynamic Up-sampling Based on Sampling As shown in Fig. 4(a), DySample first creates a sampling set S from the input feature map X using a sampling point generator. The design of the sampling point generator allows the positions of the sampling points to be dynamically determined to better capture the geometric information in the input features. Then, the `grid_sample` function is used to re-sample the input feature map X based on the position information in the sampling set S, generating the up-sampled feature map X'. This process ensures that the up-sampling can adaptively adjust according to the content of the input features, improving the quality and relevance of feature representation [12].

(2) Sampling Point Generator in DySample Fig. 4(b) details the two design approaches of the sampling point generator, based on static range factors and dynamic range factors, respectively [12].
- **Static Range Factor**: Uses a linear layer and pixel shuffle technology to generate offsets O with a fixed range factor. The offsets are added to the original grid positions G to form the sampling set S. This method provides a simple and effective way to generate sampling points, although it may not be flexible enough in some cases.
- **Dynamic Range Factor**: Introduces a dynamic range factor to increase the flexibility of generating sampling points. First, offsets O are generated using a linear layer and pixel shuffle technology. Then, a range factor is generated using the Sigmoid function σ, and finally, the offsets O are adjusted using the range factor. The introduction of the dynamic range factor allows the model to dynamically adjust the distribution of sampling points according to the specific content of the input features, more effectively capturing the geometric information in the features.

Dy-CCFM is essential for accurately identifying defects on steel surfaces, which are often difficult to distinguish from the steel's natural texture and gloss. By dynamically fusing features from multiple network depths, Dy-CCFM adaptively captures the differences between defective and normal regions, enhancing the model's ability to discriminate defects.

3.3. Minimum point distance intersection over union

MPDIoU [23] is specifically designed to enhance bounding box regression for steel surface defect detection. Precise localization is crucial for correctly identifying a wide range of defect types, including cracks, inclusions, patches, pitting, rolling scars, and scratches. Traditional loss functions such as IoU, GIoU [41], DIoU, CIoU [38], and EIoU [39] often struggle in complex scenarios, particularly when the predicted and ground-truth boxes have similar aspect ratios but differ significantly in size. To address this limitation, MPDIoU introduces a minimum point distance term, which refines the geometric relationship between bounding boxes, leading to more accurate and robust regression results.

The calculation of MPDIoU begins by determining the coordinates of the top-left and bottom-right corners of both the predicted and ground-truth boxes, as shown in Fig. 5. By incorporating the minimum point distance into the regression loss, MPDIoU ensures that even when bounding

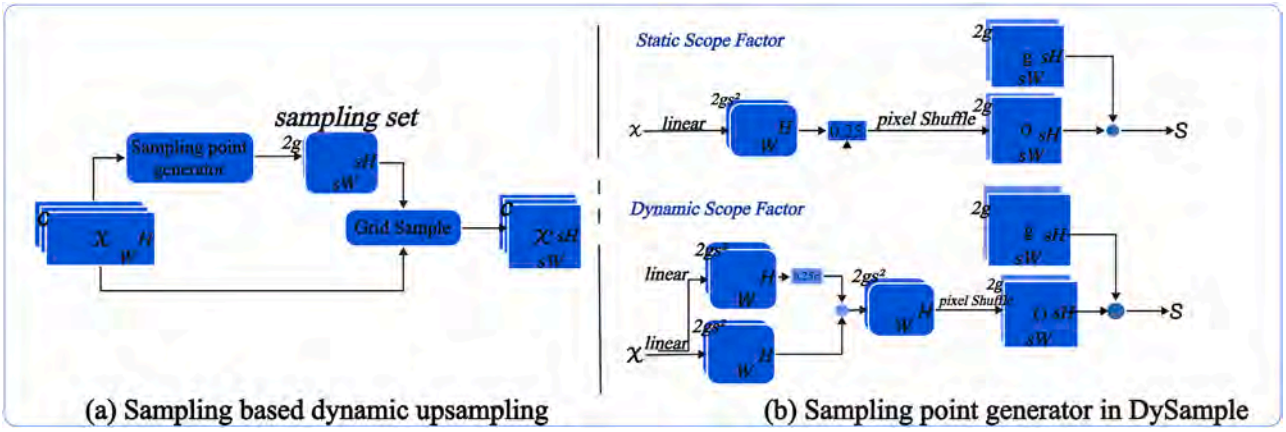


Fig. 4. Structure Diagram of the DySample Module.(a) The input feature map x is processed to generate sampling points for dynamic upsampling, and Grid Sample produces the upsampled output x'. (b) The sampling point generator consists of a static and a dynamic scope factor branch. The variables g, s, and O denote the group size, upsampling factor, and offset map, respectively. Their combination yields the final sampling scope S.

boxes are similar in shape but slightly misaligned, the loss function provides a meaningful gradient that guides the model toward better localization. This improvement is particularly valuable in industrial settings, where minor inaccuracies in defect detection can significantly impact quality control and safety.In practice, computing MPDIoU starts by determining the coordinates of the top-left and bottom-right corners of the two bounding boxes:

- The top-left coordinates of the predicted bounding box and the ground truth bounding box are $(x_{\text{pred1}}, y_{\text{pred1}})$ and $(x_{\text{gt1}}, y_{\text{gt1}})$, respectively.
- The bottom-right coordinates of the predicted bounding box and the ground truth bounding box are $(x_{\text{pred2}}, y_{\text{pred2}})$ and $(x_{\text{gt2}}, y_{\text{gt2}})$, respectively.

Subsequently, determine the intersection and union areas of the two bounding boxes:

$$A_{\text{inter}} = (x_{\text{inter2}} - x_{\text{inter1}}) \times (y_{\text{inter2}} - y_{\text{inter1}}) \tag{10}$$

$$A_{\text{union}} = A_{\text{pred}} + A_{\text{gt}} - A_{\text{inter}} \tag{11}$$

Next, compute the squared distances between the top-left and bottom-right corners:

$$d_1^2 = (x_{\text{pred1}} - x_{\text{gt1}})^2 + (y_{\text{pred1}} - y_{\text{gt1}})^2 \tag{12}$$

$$d_2^2 = (x_{\text{pred2}} - x_{\text{gt2}})^2 + (y_{\text{pred2}} - y_{\text{gt2}})^2 \tag{13}$$

Finally, the MPDIoU is calculated as:

$$\text{MPDIoU} = \frac{A_{\text{inter}}}{A_{\text{union}}} - \frac{d_1^2}{w^2 + h^2} - \frac{d_2^2}{w^2 + h^2} \tag{14}$$

where:

- A_{gt} and A_{pred} denote the areas of the ground truth and predicted bounding boxes, respectively.
- d_1 and d_2 are the distances between the top-left and bottom-right points, respectively.
- w and h represent the width and height of the input image.

MPDIoU adjusts the IoU value by subtracting the squared distances of the two minimum points, offering a more accurate geometric relationship between the predicted and ground truth boxes.

To further demonstrate the advantage of MPDIoU, Fig. 6 presents a visual comparison between IoU and MPDIoU. As shown in Fig. 6(a), conventional IoU may yield high overlap values even when the predicted box and ground-truth box differ in size but share similar aspect ratios, leading to inaccurate regression feedback. In contrast, Fig. 6(b) shows that MPDIoU effectively penalizes geometric discrepancies, achieving more accurate localization and lower regression error.

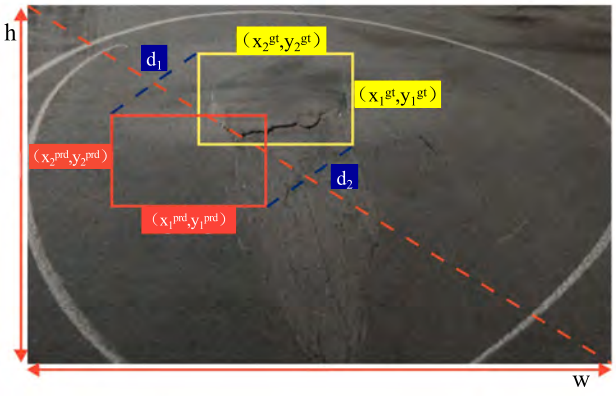


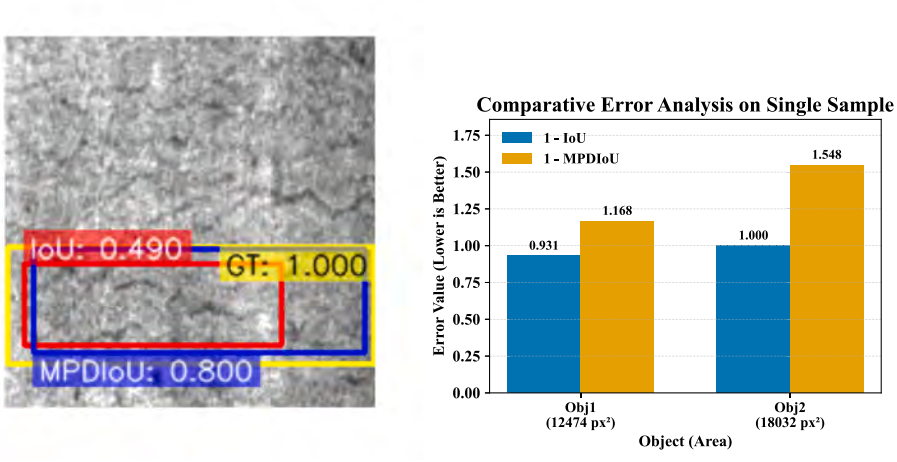
Fig. 5. Factors of MPDIoU.

4. Experiments

4.1. Dataset

In this study, we utilized two publicly available steel surface defect detection datasets: the Northeastern University Surface Defect Detection Dataset (NEU-DET) curated by Ke Chen's team at Northeastern University, and the GC10-DET dataset compiled by Tianjin University. The NEU-DET dataset served as our primary experimental resource. It comprises 1800 grayscale images, each measuring 200 pixels, evenly distributed across six defect categories: crazing (CR), inclusion (IN), patches (PA), pitted surface (PS), rolled-in scale (RS), and scratches (SC), with 300 images per category. Each image is accompanied by corresponding annotation information, making this dataset exceptionally valuable for training and evaluating surface defect detection algorithms. Its high diversity and complexity render it particularly significant in the research of surface defect detection.

Additionally, we employed the GC10-DET dataset to assess the generalization capabilities of our model. This dataset contains 2294 grayscale images annotated with bounding boxes, encompassing ten defect categories: punching hole (Pu), welding line (Wl), crescent gap (Cg), water spot (Ws), oil spot (Os), silk spot (Ss), inclusion (In), rolled pit (Rp), crease (Cr), and waist folding (Wf). The images were collected in industrial environments, providing a diverse set of real-world defect scenarios.



(a) IoU failure case (b) Improved localization with MPDIoU

Fig. 6. Comparison between IoU and MPDIoU. (a) IoU failure case for objects with similar aspect ratios but different sizes. (b) MPDIoU provides more accurate localization and lower regression error in such scenarios.

Table 1
Experiment setup.

Experiment setup	Version
Hardware accelerator	NVIDIA GeForce RTX 4090
PyTorch	1.2.1
CUDA	11.3
Python	3.9.7

Table 2
Evaluation of different algorithms on the NEU-DET dataset.

Data	Model	mAP_{50}	mAP_{50-95}	$FPS_{bs=12}$	Param	GFLOPs	F1
	SSD [5]	0.672	0.322	–	25.5M	75.7G	–
	Faster R-CNN [9]	0.690	0.453	–	129M	258.6G	–
	DETR [6]	0.701	0.462	–	159M	306G	–
	DANet [42]	0.685	0.560	–	10.9M	25.6G	–
	RT-DETRv1-L [17]	0.742	0.522	31	32M	110G	0.74
	RT-DETRv2-L [43]	0.745	0.523	32	31M	108G	0.75
NEU-DET	YOLOv5 [15]	0.716	0.501	49	1.9M	4.5G	0.72
	YOLOv6-3.0 [16]	0.710	0.497	41	4.7M	4.7G	0.67
	YOLOv8 [18]	0.730	0.512	51	3.2M	8.7G	0.73
	Gold-YOLO [44]	0.733	0.519	44	5.6M	12.1G	0.74
	YOLOv9 [19]	0.737	0.521	46	2.2M	9.6G	0.75
	YOLOv10 [20]	0.741	0.524	58	2.7M	8.4G	0.76
	Dynamic-YOLO	0.751	0.541	61	1.6M	6.0G	0.81

For both datasets, we partitioned the data into training, validation, and test sets using a 7:2:1 ratio. Specifically, for the NEU-DET dataset, we randomly selected 1260 images for training, 360 for validation, and 180 for testing, ensuring no overlap between the sets. The GC10-DET dataset was partitioned in the same manner as the NEU-DET dataset.

The proposed model was implemented using the Ultralytics YOLO framework. All experiments were conducted on a single NVIDIA GPU. The input image size was set to 640×640, and training was performed for 300 epochs with a batch size of 4. The SGD optimizer was employed, while automatic mixed precision (AMP) was enabled to improve computational efficiency. The training process used the default YOLO data augmentation strategies, and mosaic augmentation was enabled except in the final epochs.

Table 1 outlines the experimental configuration employed in this study.

4.2. Comparative experiment

In this study, we compared the performance of various object detection models on two steel surface defect datasets—NEU-DET and GC10-DET—with the results presented in Tables 2 and 3, respectively.

Table 3
Evaluation of different algorithms on the GC10-DET dataset.

Data	Model	mAP_{50}	mAP_{50-95}	F1
	RT-DETRv1-L [16]	0.730	0.537	0.70
	RT-DETRv2-L [39]	0.733	0.539	0.73
	YOLOv5 [14]	0.715	0.499	0.70
	YOLOv6 [15]	0.707	0.481	0.67
GC10-DET	YOLOv8 [17]	0.725	0.507	0.72
	Gold-YOLO [43]	0.728	0.518	0.73
	YOLOv9 [18]	0.736	0.527	0.74
	YOLOv10 [19]	0.740	0.532	0.75
	Dynamic-YOLO	0.765	0.545	0.78

Table 2 shows the performance of each model on the NEU-DET dataset. Traditional detection models such as SSD [5], Faster R-CNN [9], DETR [6], and DANet [42] exhibited moderate detection accuracy but suffered from large parameter sizes and high computational complexity. The RT-DETR series models (RT-DETRv1-L and RT-DETRv2-L) achieved some improvements in accuracy and speed [17,40]; however, their parameter sizes and computational costs remained relatively high.

The YOLO series models demonstrated progressively better detection performance with each version upgrade. Specifically, YOLOv5, YOLOv6-3.0, and YOLOv8 achieved mAP50 values ranging from 0.716 to 0.730 and inference speeds between 41 and 51 FPS. Our proposed Dynamic-YOLO model achieved the best performance on the NEU-DET dataset, with an mAP50 of 0.751, mAP50-95 of 0.541, an F1 score of 0.81, inference speed reaching 61 FPS, a parameter size of only 1.6M, and 6.0 GFLOPs. This indicates that Dynamic-YOLO maintains high accuracy while achieving faster inference speed and a smaller model size.

Table 3 presents the performance of each model on the GC10-DET dataset. It can be observed that Dynamic-YOLO also achieved the best detection results on this dataset, with an mAP50 of 0.765, mAP50-95 of 0.545, and an F1 score of 0.78. Compared to other models, Dynamic-YOLO shows significant advantages in detection accuracy and generalization capability.

To further validate the deployment advantages of Dynamic-YOLO, we measured the inference latency of various models on four representative hardware platforms-high-performance GPUs, edge GPUs, embedded AI accelerators, and general-purpose CPUs. As summarized in Table 4, Dynamic-YOLO consistently achieves the lowest inference times across all platforms, maintaining high responsiveness even on resource-constrained devices such as the Jetson Xavier NX, thereby demonstrating excellent deployment adaptability.

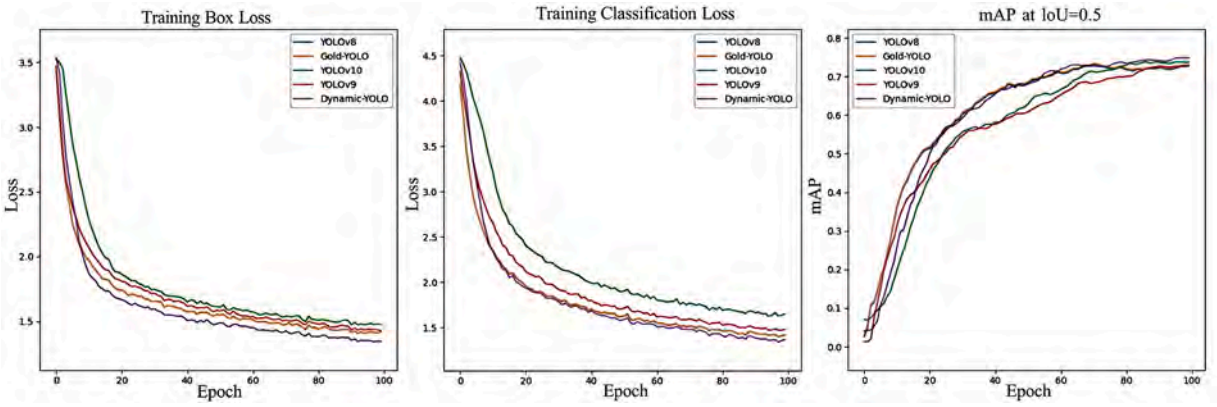


Fig. 7. Figure shows how Box Loss, Classification Loss, and mAP@0.5 evolve across epochs for various YOLO versions during training.

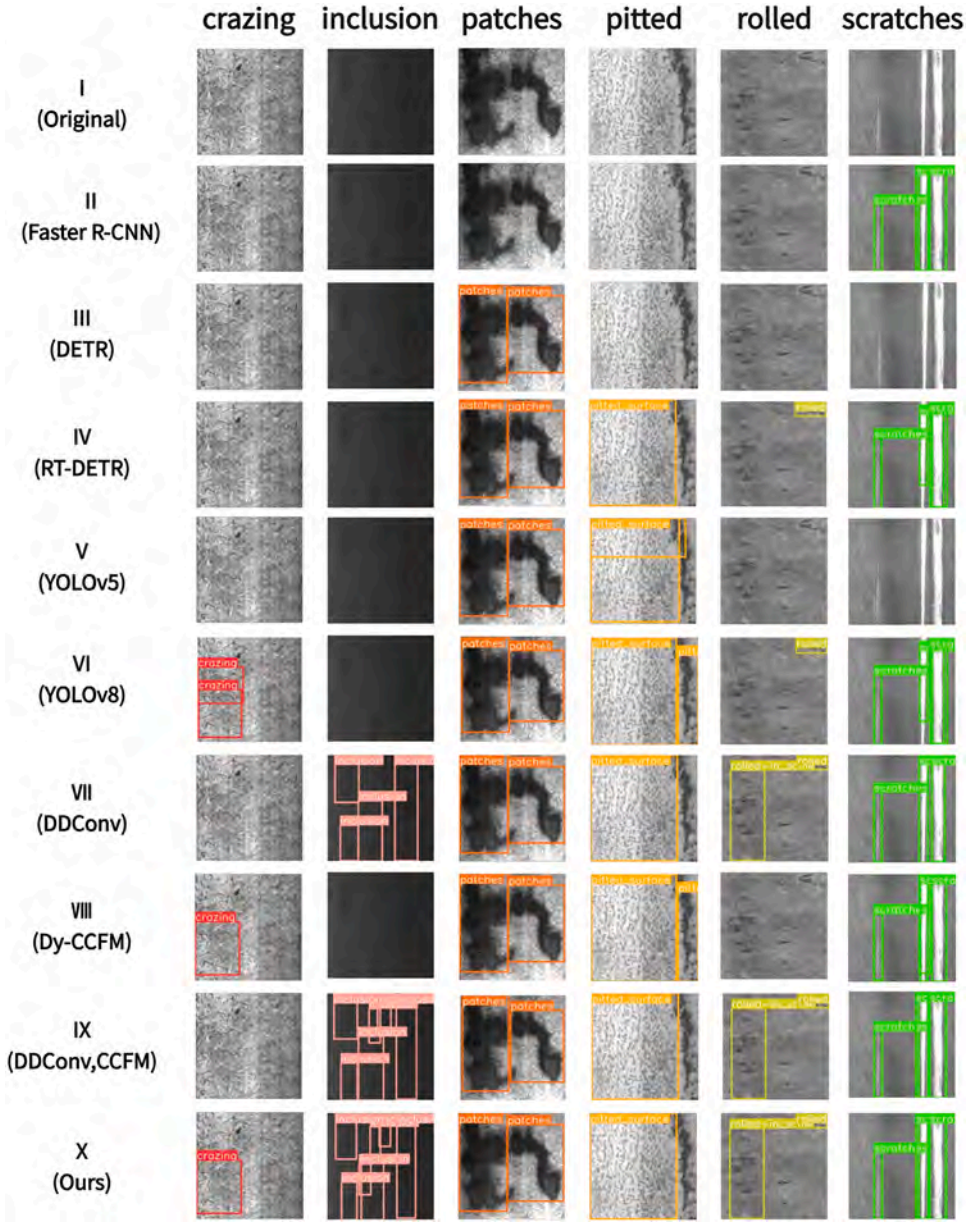


Fig. 8. Results for each method (shown in brackets) are presented, with the final row depicting dataset-label visualizations.

Table 4
Comparison of inference time (ms) across different models and hardware platforms.

Model	GPU (RTX 3090)	Edge GPU (Xavier NX)	Embedded (Ascend 310)	CPU (i7-11800H)
YOLOv8	19.61 ms	31.25 ms	35.71 ms	58.82 ms
YOLOv9	21.74 ms	33.33 ms	38.46 ms	62.50 ms
YOLOv10	17.24 ms	27.78 ms	33.33 ms	52.63 ms
RT-DETRv1	32.26 ms	52.63 ms	62.50 ms	90.91 ms
Dynamic-YOLO (Ours)	**16.39 ms**	**26.32 ms**	**31.25 ms**	**47.62 ms**

Table 5
Experimental results of different mechanism combinations.

MPDIoU	Dy-CCFM	DDConv	CR	IN	PA	PS	RS	SC	mAP_{50}
			0.412	0.694	0.889	0.904	0.620	0.864	0.730
✓			0.412	0.696	0.892	0.906	0.621	0.864	0.732
	✓		0.431	0.704	**0.900**	0.886	0.616	0.879	0.736
		✓	**0.661**	0.606	0.853	0.898	0.558	0.848	0.738
✓	✓		0.504	0.813	0.870	0.824	0.544	0.877	0.739
✓		✓	0.602	0.631	0.876	0.908	0.561	0.881	0.743
	✓	✓	0.508	**0.737**	0.878	0.909	0.613	0.840	0.747
✓	✓	✓	0.480	0.732	0.853	**0.909**	**0.661**	**0.872**	**0.751**

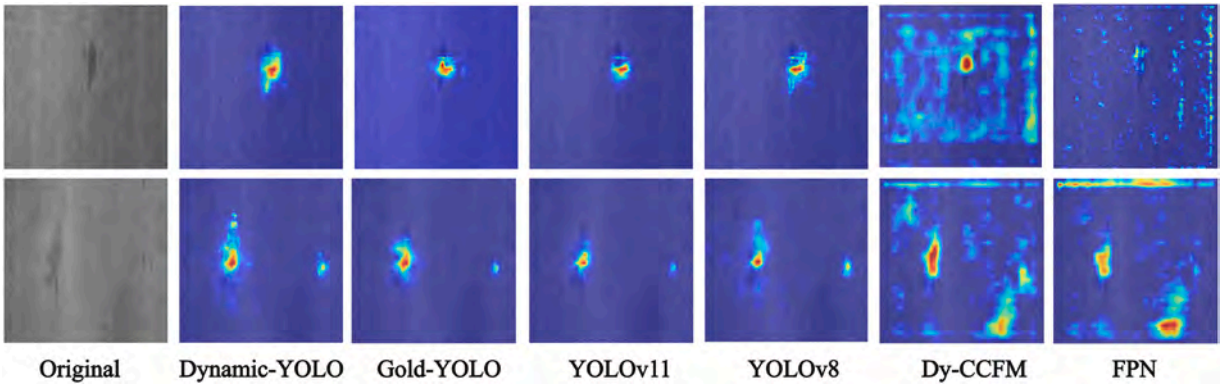


Fig. 9. Heatmap visualization results across different models.

4.3. Ablation experiments

In the ablation study detailed in Table 5, we employed YOLOv8 as the baseline, which attained an mAP_{50} of 0.730 on the NEU-DET dataset. Introducing MPDIoU alone increased the mAP_{50} to 0.732, with localization precision for Patches (PA) and Pitted Surface (PS) improving by 0.3 % and 0.2 %, respectively. Incorporating only Dy-CCFM yielded an mAP_{50} of 0.736, notably enhancing detection for Crazing (CR) from 0.412 to 0.431 and for Inclusion (IN) from 0.694 to 0.704. When solely employing DDConv, the mAP_{50} rose to 0.738, driven primarily by a dramatic 24.9 % gain on CR (from 0.412 to 0.661), although performance on IN and PA experienced slight declines.

Pairwise combinations of the proposed modules further boosted performance: MPDIoU + Dy-CCFM achieved 0.739, MPDIoU + DDConv reached 0.743, and Dy-CCFM + DDConv rose to 0.747. The MPDIoU + Dy-CCFM combination delivered the greatest improvement on IN (+11.9 %), MPDIoU + DDConv provided minor gains on PS (+0.4 %), and Dy-CCFM + DDConv enhanced both IN and PS.

When all three mechanisms were activated simultaneously, Dynamic-YOLO achieved its highest mAP_{50} of 0.751-a 2.1-point increase over the baseline. This full integration also substantially improved performance on Rolled-in Scale (RS) from 0.620 to 0.661 and on PS from 0.904 to 0.909.

Despite these gains, certain challenging conditions-such as highly reflective surfaces or severe occlusions affecting RS-continue to incur false negatives and false positives. Future work should explore illumination-invariant features or multi-modal data fusion to further enhance robustness under these extreme scenarios.

Table 6
Experimental results for diverse convolutional combinations using YOLOv8.

Model	mAP_{50}	mAP_{50-95}	Param	GFLOPs
RFAConv [45] + YOLOv8	0.731	0.518	3.03M	8.4G
DynamicConv [46] + YOLOv8	0.725	0.516	4.74M	**7.0G**
WTConv [47] + YOLOv8	0.728	0.511	2.62M	7.2G
LDConv [48] + YOLOv8	0.733	0.514	2.86M	8.0G
DDConv + YOLOv8	**0.738**	**0.522**	**2.50M**	7.1G

In addition, we conducted a comparative study by combining various types of convolutions (Conv) with YOLOv8 to evaluate their effectiveness on the NEU-DET dataset. The experimental results are shown in Table 6, where we compare our proposed DDConv with other popular convolution types. As seen in the table, DDConv combined with YOLOv8 outperforms other convolution types in both mAP50 and mAP50-95, achieving values of 0.738 and 0.522, respectively, surpassing RFAConv (0.731), DynamicConv (0.725), WTConv (0.728), and LDConv (0.733). In terms of model parameters and computational cost, DDConv has the fewest parameters, with only 2.50M parameters and 7.1G GFLOPs. In contrast, other convolution models have higher parameter counts and computational costs, with DynamicConv having 4.74M parameters and 7.0G GFLOPs.

Finally, we evaluated the effectiveness of our proposed Dy-CCFM. The experimental results of different FPN combinations with YOLOv8 are shown in Table 7. As seen in the table, Dy-CCFM combined with YOLOv8 achieves the highest performance with an mAP50 of 0.736 and mAP50-95 of 0.518, surpassing other FPN models such as RepGFPN

Table 7
Experimental results for diverse FPN combinations using YOLOv8.

Model	mAP_{50}	mAP_{50-95}	Param	GFLOPs
RepGFPN [49] + YOLOv8	0.732	0.515	3.29M	8.5G
HSFPN [50] + YOLOv8	0.730	0.513	1.96M	7.0G
BiFPN [33] + YOLOv8	0.734	0.515	2.78M	8.2G
CCFM [17] + YOLOv8	0.721	0.499	1.97M	6.7G
Dy-CCFM + YOLOv8	**0.736**	**0.518**	**1.96M**	**6.7G**

(0.732), HSFPN (0.730), BiFPN (0.734), and CCFM (0.721). Despite having the same number of parameters (1.96M) and GFLOPs (6.7G) as CCFM, Dy-CCFM achieves superior detection accuracy, demonstrating its effectiveness in improving performance.

4.4. Visualization and analysis

In this section, we present the experimental results on the NEU-DET dataset with the corresponding performance curves. We conducted an in-depth analysis of the small-object detection task for steel surface defects. As shown in Fig. 6, the first subfigure demonstrates that the proposed MPDIoU strategy significantly accelerates the convergence speed of the bounding box regression loss. This indicates that the strategy offers remarkable advantages in improving the model ability to localize target bounding boxes. Next, the second subfigure illustrates the convergence trend of the classification loss. Despite the smaller parameter size of our model (which generally leads to weaker learning capacity, slower convergence speed, and slower loss reduction), the proposed feature extraction methods enable the classification loss to decrease more rapidly. Finally, the third subfigure displays the precision curves (mAP@IoU = 0.5). It is evident that our model demonstrates effective improvements in precision, showcasing the effectiveness of the proposed approach.

Figs. 7 and 8 complement Fig. 6 by showing detection results. In addition, Fig. 9 provides the heatmaps across models, highlighting differences in feature attention and localization.

5. Discussion

5.1. Comparison with latest YOLO versions

Although this study primarily compares Dynamic-YOLO with YOLOv5-YOLOv10, the YOLO series has continued to advance beyond YOLOv13. Recent versions-particularly YOLOv13 and the latest-generation YOLO26-introduce substantial improvements in backbone design, decoupled detection heads, and multi-scale feature interaction mechanisms. YOLO26 further strengthens these advancements through redesigned hierarchical feature aggregation, enlarged receptive-field convolutions, and optimized lightweight architectures, achieving stronger performance on dense and small-object detection tasks.

Despite these upgrades, the core challenges in industrial surface defect detection-such as extremely small defect scales, low contrast between defects and background, and interference from complex textures-remain insufficiently addressed by general-purpose YOLO models. In contrast, the modules proposed in Dynamic-YOLO, including DDConv for efficient downsampling, Dy-CCFM for adaptive multi-scale fusion, and MPDIoU for accurate localization, directly target these domain-specific issues. Because these modules follow a fully modular design, they can be seamlessly integrated into newer architectures such as YOLOv13 and YOLO26 without structural conflict.

To further validate the adaptability and scalability of Dynamic-YOLO, future work will involve embedding these modules into YOLO26 and evaluating their performance under the latest YOLO paradigms. This will provide a more comprehensive understanding of Dynamic-YOLO's compatibility with state-of-the-art detectors.

5.2. Advantages and limitations in application

From an application perspective, the proposed Dynamic-YOLO demonstrates strong adaptability and robustness in detecting small and complex defects under real-world industrial conditions. Its lightweight design facilitates deployment on edge devices and real-time inspection systems with limited computational resources. In practical steel production environments, the model effectively distinguishes subtle defects from normal surface textures, providing a reliable basis for automated quality control and defect management.

Nevertheless, as with most deep learning-based approaches, its performance still relies heavily on the diversity and representativeness of the training data. Insufficient or biased datasets may hinder generalization when applied to different materials, lighting conditions, or imaging systems. In future research, we plan to expand the dataset and incorporate transfer learning and domain adaptation techniques to improve model robustness across varying industrial scenarios. Moreover, integrating temporal information from video sequences could further enhance defect tracking stability and reduce false detections in dynamic monitoring environments.

6. Conclusions

In this paper, we introduce Dy-CCFM, a dynamic feature fusion mechanism designed to address the challenge that steel surface defects often closely resemble the underlying texture and gloss. Dy-CCFM not only markedly enhances model compactness but also ensures that more discriminative features are delivered to the detection head. Concurrently, we develop DDConv, an efficient and lightweight convolutional module that reduces channel dimensionality while preserving critical information and providing finer-grained feature representations. Together, these modules enable the proposed Dynamic-YOLO to achieve a competitive balance of minimal parameter footprint and computational load, attaining a peak detection accuracy of 75.1

Nonetheless, Dynamic-YOLO exhibits limitations in certain extreme scenarios. Under extensive specular highlights (e.g., strong mirror-like reflections) or heavy oil contamination, Dy-CCFM's ability to distinguish fine cracks from background textures degrades. In cases of severe occlusion or overlapping defects (for instance, pits co-occurring with scratches), the model may erroneously merge distinct defect types. Furthermore, on very low-resolution inputs or under highly uneven illumination, DDConv can lose critical detail during feature reconstruction. Therefore, the applicability of Dynamic-YOLO is best confined to routine industrial inspection environments. Future work should consider integrating illumination-invariant or multi-modal (e.g., depth or thermal) cues, augmenting the training set with extreme-condition samples, and refining Dy-CCFM's adaptive weighting strategy to bolster robustness against complex interferences and extreme defect presentations.

Data availability

Data will be made available on request.

CRediT authorship contribution statement

Li-Juan Liu: Writing - original draft, Supervision, Software, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization; **Shao-Qi Sun:** Writing - original draft, Visualization, Validation, Software, Methodology, Formal analysis, Data curation, Conceptualization; **Hamid Reza Karimi:** Writing - original draft, Visualization, Supervision, Resources, Methodology, Investigation, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

The work was supported by the European Union's Horizon Europe programme under the Marie Sklodowska-Curie grant agreement no. 101073037.

References

[1] Z.Q. Zhao, P. Zheng, S.T. Xu, X. Wu, Object detection with deep learning: a review, IEEE Trans. Neural Netw. Learn. Syst. 30 (11) (2019) 3212–3232.
[2] Y. He, X. Wang, Y. Chen, Steel surface defect recognition with a new-oriented deep learning method, IEEE Access 6 (2018) 10240–10252.
[3] F. Chen, X. Liu, H. Zhang, Y. Luo, N. Lu, Y. Liu, X. Xiao, Assessment of fatigue crack propagation and lifetime of double-sided U-rib welds considering welding residual stress relaxation, Ocean Eng. 332 (2025) 121400.
[4] Y. Lecun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (2015) 436–444.
[5] W. Liu, et al., SSD: single shot multibox detector, in: Proceedings of the European Conference on Computer Vision (ECCV), 2016, pp. 21–37.
[6] N. Carion, et al., End-to-end object detection with transformers, in: Proceedings of the European Conference on Computer Vision (ECCV), 2020, pp. 213–229.
[7] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: unified, real-time object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 779–788.
[8] R. Girshick, Fast R-CNN, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1440–1448.
[9] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks, IEEE Trans. Pattern Anal. Mach. Intell. 39 (6) (2017) 1137–1149.
[10] L. Zhang, Q. Zhou, X. Wu, W. Liu, Automatic surface defect detection for cold-rolled steels using attention mechanism based deep learning, Ultrasonics 96 (2019) 22–31.
[11] X. Kou, S. Liu, K. Cheng, Y. Qian, Development of a YOLO-V3-based model for detecting defects on steel strip surface, Measurement 182 (2021).
[12] H. Peng, et al., Dynamic sampling networks for efficient image recognition, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 517–526.
[13] L.J. Liu, Y. Zhang, H.R. Karimi, Resilient machine learning for steel surface defect detection based on lightweight convolution, Int. J. Adv. Manuf. Technol. 134 (2024) 4639–4650. https://doi.org/10.1007/s00170-024-14403-z
[14] L. Liu, Y. Zhang, H.R. Karimi, Defect detection of printed circuit board surface based on an improved YOLOv8 with FasterNet backbone algorithms, in: Signal, Image and Video Processing, 2024.
[15] G. Jocher, YOLOv5 by Ultralytics (Version 7.0) [Computer software], 2020. https://doi.org/10.5281/zenodo.3908559
[16] C. Li, et al., YOLOv6: a single-stage object detection framework for industrial applications, 2022. https://github.com/meituan/YOLOv6.
[17] C.Y. Wang, A. Bochkovskiy, 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, 2023, pp. 7464–7475.
[18] G. Jocher, J. Qiu, A. Chaurasia, Ultralytics YOLO (Version 8.0.0), 2023. https://github.com/ultralytics/ultralytics.
[19] C.-Y. Wang, H.-Y.M. Liao, YOLOv9: learning what you want to learn using programmable gradient information, in: Proceedings of the European Conference on Computer Vision (ECCV), Springer, 2024.
[20] A. Wang, H. Chen, L. Liu, K. Chen, Z. Lin, J. Han, G. Ding, Real-time end-to-end object detection, YOLOv 10 (2024) 107984–108011.
[21] B. Liu, D. Chen, X. Qi, YOLO-pdd: a novel multi-scale PCB defect detection method using sequential image depth representations, in: Proceedings of the International Conference on Neural Information Processing (ICONIP), Springer, 2024, pp. 297–313.
[22] T. Zhang, H. Pang, C. Jiang, GDM-YOLO: a model for steel surface defect detection based on YOLOv8s, IEEE Access 12 (2024) 148817–148825. https://doi.org/10.1109/ACCESS.2024.3476908
[23] J. Chen, X. Zhang, Y. Li, Enhancing YOLO for multi-scale object detection in industrial applications, IEEE Trans. Ind. Electron. 71 (6) (2023) 1123–1134.
[24] S. Ma, Y. Xu, MPDIoU: a loss for efficient and accurate bounding box regression, arXiv preprint arXiv:2307.07662, 2023.
[25] Y. Himeur, S. Al-Maadeed, H. Kheddar, et al., Video surveillance using deep transfer learning and deep domain adaptation: towards better generalization[J], Eng. Appl. Artif. Intell. 119 (2023) 105698.
[26] D. Li, et al., Involution: inverting the inherence of convolution for visual recognition, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 12321–12330.
[27] L. Qi, Distribution shifted convolution: a deep learning operator for object segmentation, Proceedings of the AAAI Conference on Artificial Intelligence 35 (2021) 2857–2865.
[28] X. Zhang, et al., Spatial attention mechanisms in convolutional neural networks: a survey, IEEE Access 8 (2020) 135361–135376.
[29] S. Woo, J. Park, J.-Y. Lee, I.S. Kweon, CBAM: convolutional block attention module, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 3–19.
[30] Q. Hou, D. Zhou, 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. 13713–13722.
[31] M.T. Islam, K.M. Hasib, M.M. Rahman, et al., Convolutional auto-encoder and independent component analysis based automatic place recognition for moving robot in invariant season condition. Human-Centr. Intell. Syst., 3 (1), 2023 13–24.
[32] S. Liu, D.J. Lin, Y.J. Xie, Path aggregation network for instance segmentation, in: CVPR, 2018.
[33] M. Tan, R. Pang, Q.V. Le, EfficientDet: scalable and efficient object detection, in: CVPR, 2020.
[34] S. Huang, W. Zhong, Y. Liu, FaPN: feature-aligned pyramid network for dense image prediction, in: ICCV, 2021.
[35] A. Thakkar, R. Lohiya, Fusion of statistical importance for feature selection in deep neural network-based intrusion detection system, Inf. Fusion 90 (2023) 353–363. C, https://doi.org/10.1016/j.inffus.2022.09.026
[36] B. Jiang, J. Li, H. Li, R. Li, D. Zhang, G. Lu, Enhanced frequency fusion network with dynamic hash attention for image denoising, Inf. Fusion 92 (2023) 420–434. C, https://doi.org/10.1016/j.inffus.2022.12.015
[37] Z. Xin, S. Chen, T. Wu, Y. Shao, Few-shot object detection: research advances and challenges, Inf. Fusion 107 (2024). C, https://doi.org/10.1016/j.inffus.2024.102307
[38] Z. Zheng, et al., Distance-IoU loss: faster and better learning for bounding box regression, Proceedings of the AAAI Conference on Artificial Intelligence 34 (2020) 12993–13000.
[39] B. Zhang, et al., Focal and efficient IOU loss for accurate bounding box regression, Neurocomputing 506 (2022) 146–157.
[40] W. Liu, H. Lu, H. Fu, Z. Cao, Learning to upsample by learning to sample, in: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2023, pp. 6027–6037.
[41] Q. Wu, NEU-DET, IEEE Dataport 2024. https://doi.org/10.21227/j84r-f770
[42] F. Zhao, Y. Zheng, Y. Zhang, DANet: dual attention network for scene segmentation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 3146–3154. https://doi.org/10.1109/ICCV.2019.00321
[43] W. Lv, Y. Zhao, Q. Chang, K. Huang, G. Wang, Y. Liu, RT-DETRv2: Improved Baseline with Bag-of-Freebies for Real-Time Detection Transformer, abs/2407.17140, 2024. https://api.semanticscholar.org/CorpusID.
[44] C. Wang, W. He, Y. Nie, J. Guo, C. Liu, K. Han, Y. Wang, Gold-YOLO: efficient object detector via gather-and-distribute mechanism, Adv. Neural Inf. Process. Syst. 36 (2023).
[45] X. Zhang, C. Liu, D. Yang, T. Song, Y. Ye, K. Li, Y. Song, in: RFAConv: Innovating Spatial Attention and Standard Convolutional Operation, 2023. https://arxiv.org/abs/2304.03198.
[46] K. Han, Y. Wang, J. Guo, E. Wu, ParameterNet: parameters are all you need for large-scale visual pretraining of mobile networks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2024, pp. 15751–15761.
[47] S.E. Finder, R. Amoyal, E. Treister, O. Freifeld, Wavelet convolutions for large receptive fields, in: European Conference on Computer Vision, 2024.
[48] X. Zhang, Y. Song, T. Song, D. Yang, Y. Ye, J. Zhou, L. Zhang, LDConv: linear deformable convolution for improving convolutional neural networks, Image and Vision Computing, 105190, Elsevier, 2024.
[49] X. Xu, Y. Jiang, W. Chen, Y. Huang, Y. Zhang, X. Sun, DAMO-YOLO: a report on real-time object detection design, arXiv preprint arXiv:2211.15444v2, 2022.
[50] Y. Chen, C. Zhang, B. Chen, Y. Huang, Y. Sun, C. Wang, X. Fu, Y. Dai, F. Qin, Y. Peng, Y. Gao, Accurate leukocyte detection based on deformable-DETR and multi-level feature fusion for aiding diagnosis of blood diseases, Comput. Biol. Med. 170 (2024) 107917. https://doi.org/10.1016/j.compbiomed.2024.107917