Knowledge-Based Systems 260 (2023) 110176

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Progressive refined redistribution pyramid network for defect detection in complex scenarios

Xuyi Yu [a], Wentao Lyu [a,*], Chengqun Wang [a], Qing Guo [b], Di Zhou [c], Weiqiang Xu [a]

[a] Key Laboratory of Intelligent Textile and Flexible Interconnection of Zhejiang Province, Zhejiang Sci-Tech University, Hangzhou, 310018, Zhejiang, China
[b] Zhejiang Technical Innovation Service Center, Hangzhou, 310007, Zhejiang, China
[c] Zhejiang Uniview Technologies Co., Ltd., Hangzhou, 310051, Zhejiang, China

ARTICLE INFO

Article history:
Received 3 June 2022
Received in revised form 30 November 2022
Accepted 4 December 2022
Available online 7 December 2022

Keywords:
Defect detection
FPN
Feature alignment
Supervision
YOLOv5
Complex scenarios

ABSTRACT

Due to the rapid development of manufacturing capabilities and the general improvement in requirements for product quality, the role of quality inspection in the industrial production process is becoming increasingly important. Unlike the case for natural objects, detailed information is particularly crucial in defect classification and localization, resulting in poor performance of general object detectors on complex defect detection tasks. Therefore, this paper proposes a progressively refined redistribution pyramid network for visual defect detection in complex images, in which three novel components are designed. (1) The aligned dense feature pyramid network (AD-FPN) refines scale differences and performs efficient alignment, alleviating feature misalignment in FPN-based methods. (2) The phase-wise feature redistribution module (PFRM) enhances the interaction between features across layers, where global information is reassigned in a semantically adaptive manner. (3) The adaptive feature purification module (AFPM) helps the network distinguish defects from complex backgrounds, and its update is directly supervised by an auxiliary branch to accelerate convergence. These ideas are all implemented based on YOLOv5. Extensive experiments on the Tianchi fabric dataset, the publicly available surface defect dataset NEU-DET, and the PCB defect dataset show that our method outperforms other state-of-the-art defect detection methods on most evaluation metrics. In addition, experimental results on the remote sensing dataset RSOD and pothole image dataset also demonstrate the strong generalization ability of our method in other complex scenarios.

1. Introduction

Object detection is an important task in computer vision and is widely used in many fields, such as surface defect detection [1–3], remote sensing [4], medical diagnosis [5] and real-time vehicle detection in autonomous driving scenarios [6], etc. Different from the regularity of natural objects, the defects in industrial products have various shapes and sizes, which brings much trouble to quality inspectors.

In recent years, some efficient defect detection methods based on deep learning have also been proposed [2,7,8], aiming to achieve efficient automatic detection to reduce labor costs. However, most of them focus on the quantity or quality of extracted features and ignore the utilization of the relationships between multi-scale features. Due to the scale gap between multi-scale features, the existing semantic mismatch problem will cause the boundary features to be submerged in the fusion process. Note that the boundary information of defects is crucial for classification and localization, so it is necessary to obtain complete and effective features of boundary to achieve accurate defect detection. In addition, in some specific defect detection scenarios such as fabric defect detection, the complex textures may cause serious interference with non-salient defects, and the same problem also exists in the object detection of remote sensing images. We list some non-salient defects (small to medium scale) in fabric images, as shown in Fig. 1.

Most of the existing defect detection methods are developed based on general object detectors. However, due to the great variation in the shape of the defects and the diversity of application scenarios, general object detectors encounter many problems when applied to defect detection and other complex detection tasks. After analysis, we believe that the following issues are very meaningful and important for improving the detection performance.

Feature misalignment: As an important component of modern detectors, the feature pyramid network (FPN) [9] and its variants are often employed to aggregate the multi-level features output

* Corresponding author.
E-mail address: alvinlwt@zstu.edu.cn (W. Lyu).

https://doi.org/10.1016/j.knosys.2022.110176
0950-7051/

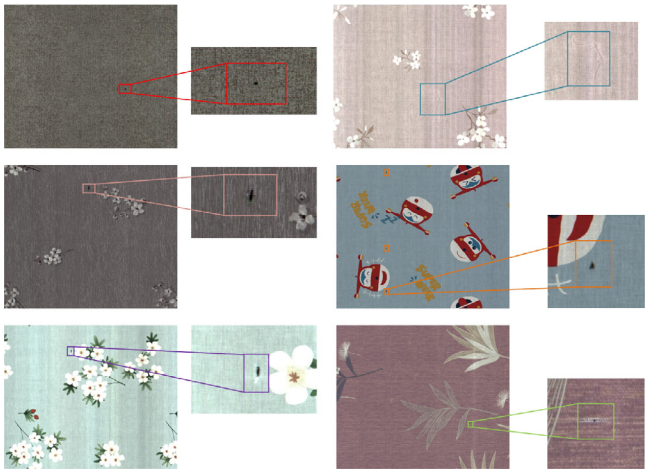

Fig. 1. Some examples of small and medium-scale defects in fabric images that are easily overwhelmed by the backgrounds.

by the backbone network. However, due to the large-scale variation between features at different semantic levels, there exists a serious pixel-level misalignment problem when they are fused with each other.

Interference from complex backgrounds: In the tasks of fabric defect detection and object detection in remote sensing images, there exists a serious problem of background interference. As a typical example, the complex textures of the fabric submerges some background-related defects. A similar situation occurs in remote sensing images due to differences in geometric morphology and light intensity.

Insufficient interaction between cross-layer features: In most vanilla FPN structures, the features of adjacent layers are gradually fused in a bottom-up or top-down manner. However, this approach will result in a lack of direct interaction between cross-layer features. This means that low-level features cannot fully use semantically rich features for classification, and high-level features cannot directly utilize the accurate localization information of low-level features to help locate targets.

In the current general object detectors, the FPN-based structures [10,11] are usually employed to aggregate features of different semantic levels, and most of them adopt a manner that fuses the features of adjacent layers sequentially. Among them, high-level features are upsampled to keep scales consistent before fusion. Since the upsampling operation is not learnable, when high-level features are upsampled and fused with local features, the objects represented by the two features at the same pixel position may be inconsistent, which is the problem of feature misalignment. It can be easily seen that the problem directly caused by the misalignment is the information loss of the object boundary, which will bias the prediction of the subsequent detection head. For the feature misalignment problem, many works [12–14] have also been proposed, all of which are based on pixel-level adjustment by deformable convolution (DCN) [15], but there is a problem of large computational complexity. Motivated by the idea of spatial transformer network (STN) [16], some lightweight alignment schemes have also been proposed in the state-of-the-art (SoTA) semantic segmentation methods [17,18], but they are generally less effective in the detection field. Different from their methods, we believe that the root cause of the feature misalignment comes from the problem of large scale gaps at different levels. The adjacent feature layers of the FPN-like structure generally have a scale span of 2 times, which will lead to a greater probability of misalignment when the features are fused with each other. Therefore, based on the original layers with a span of 2 times, we additionally generate medium-scale features to help the transition of upsampling between adjacent layers, and these additional layers are not used for final prediction to save time cost. Furthermore, we find that scale gaps still exist between transition layers, so we propose a progressive fusion accordingly. It further helps the transition layers to perform conflict-free aggregation at the original scale, while dynamically aligning upsampled features at the second stage.

For the second issue above, we find that using the attention mechanism [19,20] directly does not achieve better performance, and even results in slower convergence due to additional parameters to learn. To address this problem, we introduce the direct supervision signals for the subnetwork to accelerate convergence. Subsequently, a regional mask is generated through the subnetwork to enhance the features of the defective region while relatively weakening the influence of the background. However, through feature visualization, we find that the features of different channels can be divided into two groups according to whether they are position-sensitive. This finding illustrates that the practice of directly applying masks [21,22] will affect the position-insensitive channels, which is not perfect and elegant. Therefore, we propose a channel selection module to avoid applying regional masks to position-insensitive features, thereby further improving the ability to distinguish defects from complex backgrounds.

For the third issue above, the common practice is to introduce additional upsampling or downsampling paths to directly enhance the interaction between cross-scale features, but this will lead to pixel-level feature conflicts due to the large semantic gap. Similar to the multi-level feature aggregation methods adopted in the Balanced FPN [23] and YOLO-Ret [24], we also first perform feature aggregation to generate features containing global information. Then, in order to control the amount of computation, we use 1×1 convolution to compress the feature dimension. In particular, considering that each layer in Neck requires different levels of semantics, we propose to generate features at different semantic levels based on aggregated features in a phase-wise manner, and then assign them to different feature layers as global guidance information.

Based on the above discussion, we propose three novel modules, called Adaptive Feature Purification Module (AFPM), Aligned Dense FPN (AD-FPN) and Phase-wise Feature Redistribution Module (PFRM), respectively. By integrating these modules into state-of-the-art YOLOv5 [25], we obtain an efficient network with strong defect detection capability in complex scenarios, called CS-YOLO (YOLOv5 for Complex Scenarios). Extensive experiments on three publicly available defect datasets demonstrate the effectiveness of the proposed method. To further test the adaptability of CS-YOLO in other complex scenarios, we introduce remote sensing dataset and pothole image dataset for generalization testing.

Our contributions can be summarized as follows.

- We propose a progressively refined redistribution pyramid with supervised attention for complex defect detection, which is developed based on YOLOv5s and achieves state-of-the-art performance on multiple datasets. We mainly redesign the feature fusion network, consisting of AFPM, AD-FPN and PFRM.
- AFPM first groups the channels and then directly helps the network distinguish defects from the background by generating masks of defect regions. In particular, strong supervision is used to stabilize the update of the parameters.
- Through finer-grained division of layers and interlayer alignment, AD-FPN greatly alleviates the problem of feature misalignment in FPN-based networks, and achieves accurate representation of defect boundaries.

- By adaptively redistributing features at different stages with global information, PFRM dynamically supplements information for features at all levels, realizing a better cross-layer interaction scheme.

2. Related work

2.1. Defect detection networks

Defect detection generally refers to the detection of defects on various surfaces to ensure product quality and maintain production stability. With the rapid development of deep learning, vision-based detection technology has been widely researched and applied because of its low cost and nondestructive properties.

To achieve nondestructive testing of products, Yang et al. [1] introduced various forms of attention mechanisms and dense connections in U-Net to help better fusion, and employed Dice loss to alleviate the imbalance problem. Taking the two-stage networks Faster R-CNN [26] and Cascade R-CNN [27] as baselines, Zeng et al. [7] proposed a reference-based defect detection network, which solved the two issues of texture shift and local visual confusion, and the experimental results on Tianchi aluminum and fabric datasets have proved its effectiveness. However, due to the scarcity of template images in real-world scenarios and the inefficiency brought by the two-stage network, its practicability is limited, and there are similar problems existing in [28,29]. For photovoltaic cell defect detection, Su et al. [30] proposed a bidirectional attention feature pyramid (BAFPN) based on Faster R-CNN, which used multi-head non-local attention to calculate the weight ratio of each input node. In [31], multiple parallel atrous convolutions were introduced to exploit contextual information, and the performance was further improved by simple aggregation and redistribution of multi-scale features. Likewise, for tiny defect detection in industrial scenes, Yu et al. [32] improved the YOLO network and further discussed feature aggregation and distribution strategies. In addition, Lin et al. [2] proposed using the class activation map (CAM) generated by the classification network to guide the localization of defect regions, which was also mentioned in SSPNet [22].

Despite the remarkable performance of these methods, most of them focus on leveraging existing components to help models achieve better performance in various ways, ignoring the root cause of the poor performance of detectors on complex defect detection tasks. Therefore, we comprehensively analyze the architectural characteristics of one-stage detectors and the problems existing in complex defect detection, and propose solutions in terms of feature alignment, cross-layer interaction, and background filtering, aiming to improve the detection performance in complex scenarios by trading off cost.

2.2. Feature misalignment

Feature misalignment in computer vision usually refers to the pixel deviation between two features. Most of the related work is based on DCN [15] and STN [16]. DCN enhances the ability of Convolutional Neural Networks (CNN) to model geometric transformations, thereby better solving image recognition tasks with spatial deformation. STN was originally designed to enable the model to have better spatial invariance, that is, when the target undergoes a certain transformation, the model can still give the same correct result. Both the DCN and STN use a subnetwork to regress a set of parameters that reflect the corresponding relationships between inputs and outputs. According to the obtained parameters, they adjust the feature map at the pixel level, which coincides with the feature alignment. Therefore, a large number of works based on the two have been proposed, such as AlignDet [13], RefineDet [12], Reppoint [33], Guided Anchoring [34], AlignSeg [18], Semantic Flow [17], etc., which are applied in the fields of detection and segmentation, respectively. However, the DCN-based method also requires deformable convolution to achieve alignment after obtaining the offset map, while the STN-based method can be aligned by a parameter-free grid sampling operation after learning the offset map. In this paper, to further solve the problem of feature misalignment between transition layers that still exists in Dense FPN, we propose a progressive fusion alignment strategy for three-input nodes based on the adjustment function of STN, thereby obtaining AD-FPN.

2.3. Multi-scale feature fusion

Multi-scale feature fusion involves fusing features of different scales hierarchically to construct multi-level features that are rich in both spatial and semantic information. In the field of object detection, FPN first proposed a top-down structure to propagate semantic information from high-level layers to low-level layers, helping shallow layers locate small objects. Based on FPN, a large number of pyramid structures have been proposed. An additional bottom-up path is introduced in PANet [10] based on FPN to re-propagate the low-level localization information back to the high-level layers. NAS-FPN [35] applies the latest network architecture search (NAS) technology to search for the optimal connection mode of each node in FPN. Later, a novel BiFPN [11] structure is designed in EfficientDet, which uses learnable weights to assign the fusion ratio of features at each scale, and introduces lateral connections to strengthen internal features, surpassing NAS-FPN in the balance of accuracy and speed. A reverse feature pyramid (RevFP) is proposed in RC-Net [36], which integrates bidirectional feature fusion into fewer nodes for computation, thereby improving efficiency while maintaining the original accuracy. Some works such as ASFF [37], Balanced FPN and YOLO-ReT do not modify the connection mode of the FPN structure, but additionally perform cross-scale feature fusion before or after FPN to strengthen the information exchange between non-adjacent layers. The latest CSL-YOLO [38] proposes the generation of more prediction layers, which improves the detection ability of the model at all scales but brings an additional inference burden. In our method, AD-FPN uses a lightweight C3 module to process medium-scale features while limiting the number of residual blocks. More importantly, we still keep the number of prediction heads at 3; that is, the additional medium-scale features are used as transition layers and not used for the final prediction, which retains the inference efficiency of the network. In addition, for nodes with multiple inputs, we also innovatively propose a progressive fusion strategy to further smooth the fusion process of multi-level features.

2.4. Attention mechanism

The attention mechanism dynamically assigns weights to features in single or multiple dimensions, thus emphasizing important features and suppressing noise, which can effectively improve the representational capability of the network with limited computational cost. SENet [19] was the earliest proposed channel attention mechanism and won the championship in the classification task of ILSVRC2017. It models the relationship between channels through a bottleneck structure formed by two fully connected (FC) layers. Based on SENet, ECANet [39] proposes using fast 1D convolution to model the local relationship between channels while avoiding the information loss caused by dimensionality reduction, thereby realizing channel attention

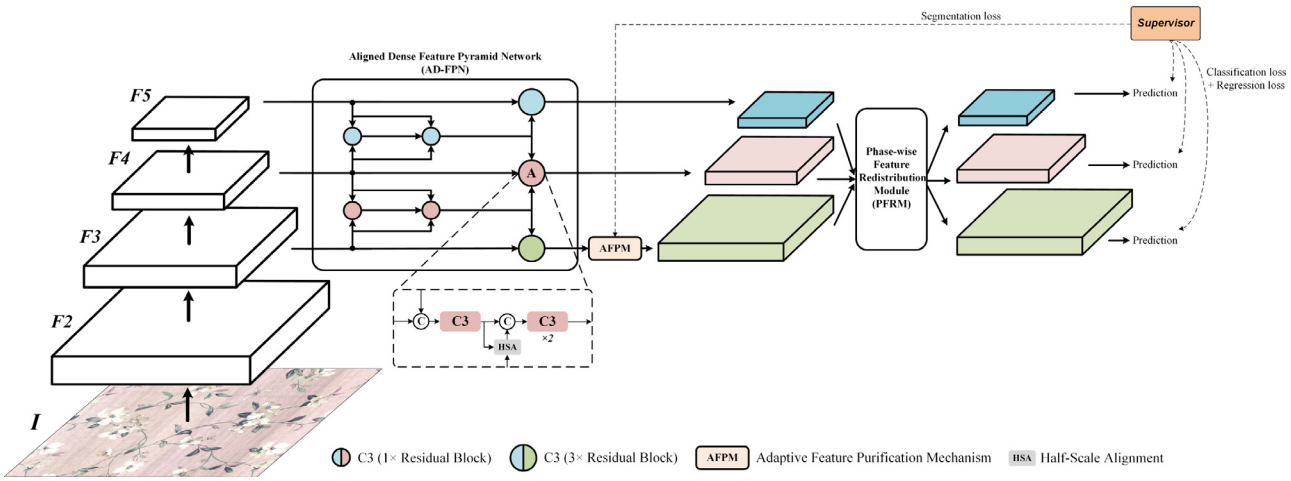

Fig. 2. Illustration of the overall architecture of the proposed CS-YOLO. The backbone network is divided into five stages, each of which is downsampled once. The following feature fusion network generates two additional medium-scale transition features based on the input features of three scales to make the feature fusion process smoother, and additionally designs a progressive fusion method for multi-input nodes to further reduce the conflict between multi-scale features. The colored filled circles represent the cross-stage partial (CSP) module, and half-scale alignment (HSA) is used to align features between transition layers. AFPM is a feature purification module, which is used to strengthen the foreground features and weaken the background noises. PFRM is designed to achieve the phase-wise redistribution of features across scales. A supervisor is employed to provide supervision signal directly to the components of the network.

at a minimal cost. CBAM [20] and BAM [40] achieve dual attention by fusing the results of spatial attention and channel attention in serial and parallel, respectively. Different from the previous methods, DA-Net [41] achieves better performance by constructing channel and spatial attention in a self-attention manner. By considering the weighting of all pixels when calculating the response of a certain position, self-attention models long-distance dependencies, but the computational complexity cannot be ignored. Transformer [42] is a further extension of self-attention and has achieved promising results in both natural language processing and vision tasks. However, these attentions are directly embedded in the network as a internal structure, and their parameters are trained together with the network, which will inevitably bring training burden and easily cause non-convergence problems. Therefore, to solve these issues, we refer to the idea of the face detection network (FAN) [21] and employ an explicit supervision signal to directly accelerate the convergence of AFPM before the attention weights are generated.

3. Method

In this section, we detail the overall design of the proposed method, as shown in Fig. 2. Like most one-stage detectors, CS-YOLO adopts the detection paradigm of Backbone+Neck+Head. We keep the backbone network of YOLOv5s as an ultra-lightweight feature extractor. In particular, we reconstruct the Neck structure and divide it into three parts for different stages of feature fusion. First, we design a feature pyramid AD-FPN with dense scales based on a divide-and-conquer approach, which weakens the differences between the original layers and further alleviates the misalignment problem. AFPM is then employed to highlight the foreground features at each scale while suppressing the noisy features brought by the background. Finally, in PFRM, we aggregate features from multiple layers and propose a progressive approach to assign features in a semantically adaptive manner. Additionally, we design an auxiliary branch to supervise the updates of AFPM explicitly.

3.1. Overall architecture

As shown in Fig. 2, we denote the feature map $I \in \mathbb{R}^{H\times W\times 3}$ as the input of the network, where H and W represent the height and width of the feature map, respectively, and 3 represents the number of RGB channels in the image. We first use a backbone network to extract multi-level features. The backbone network is divided into five stages of convolutional blocks, where each stage starts with a convolution with a stride of 2. The output features of the last three stages are selected for the subsequent fusion, which are denoted as $\{F3, F4, F5\}$, and their scales are 1/8, 1/16, 1/32 of I, respectively. $F3$, $F4$, and $F5$ are first input into the AD-FPN for bidirectional feature aggregation, and the output multi-scale features are denoted as $\{E3, E4, E5\}$. Among them, $\{E3\}$ is input to AFPM for feature enhancement of foreground, and the corresponding purified feature $\{M3\}$ is obtained. Next, $M3$, $E4$ and $E5$ are input into PFRM for further feature redistribution to strengthen the features of each level with global information, and the obtained features $\{H3, H4, H5\}$ are used for the final prediction.

3.2. Aligned dense FPN (AD-FPN)

With the rapid development of computer vision, the paradigm of the existing SoTA convolutional backbone networks such as ResNet [43] and EfficientNet [11] has been fixed, and all of them follow a multi-stage design with different feature resolutions in each stage. Similarly, the subsequent rise of the visual transformer also adopted this paradigm and achieved impressive performance. For the multi-stage features extracted from the backbone network, the detection network generally employs FPN and its variants to further fuse them before final prediction, which is proven to have better performance. However, in the fusion process of multi-stage features (such as scales of 1/8×, 1/16×, 1/32×) extracted by the backbone network, there will be a serious problem of feature misalignment due to the large scale gap.

Therefore, to better solve the misalignment problem in the FPN-like structure of the detection network, we follow the practices of CSL-YOLO [38] and adjust the structure of YOLOv5s accordingly. Furthermore, we perform feature alignment for nodes with multiple inputs, and a new FPN structure is obtained, called the aligned dense FPN (AD-FPN). The specific process will be described below.

Based on the extracted multi-stage features, we perform non-integer adaptive upsampling (1.5×) and downsampling (0.75×)

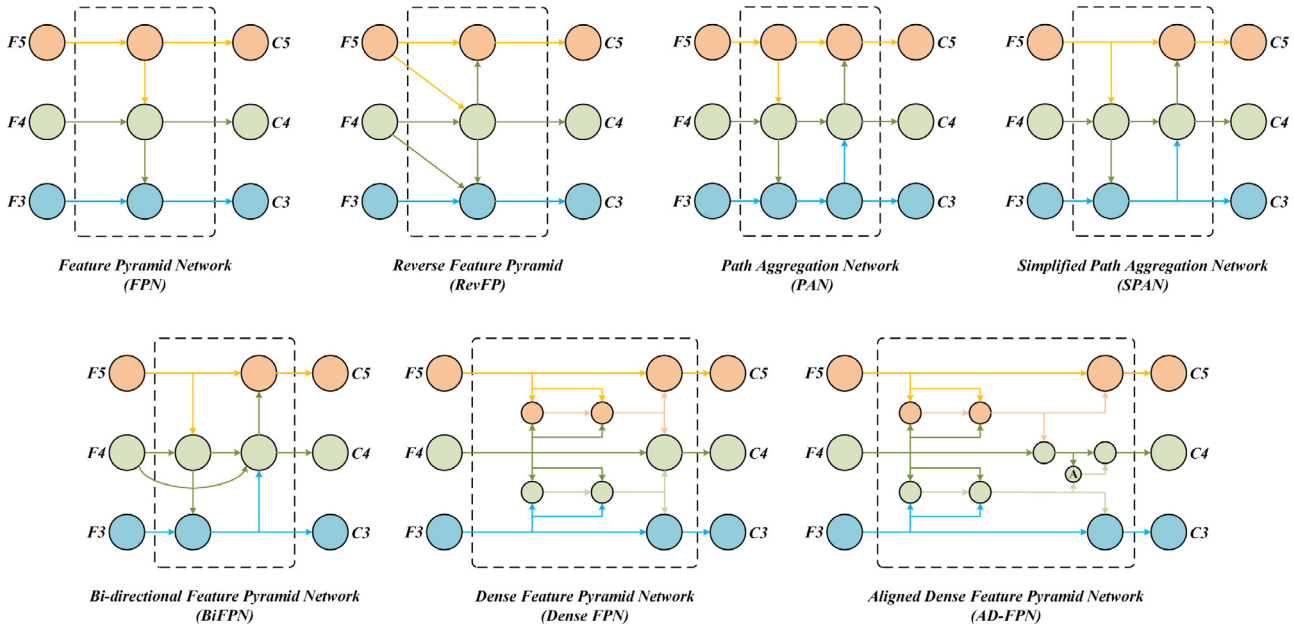

Fig. 3. Illustration of some state-of-the-art FPN-based architectures with three input layers. Dense FPN and AD-FPN are the structures proposed in this paper, in which circles with the same color represent the same number of channels, and the size of the circle represents the number of CSP modules contained (we set the small circle to represent the CSP module with one residual block, and the large circle to represent CSP with three residual blocks).

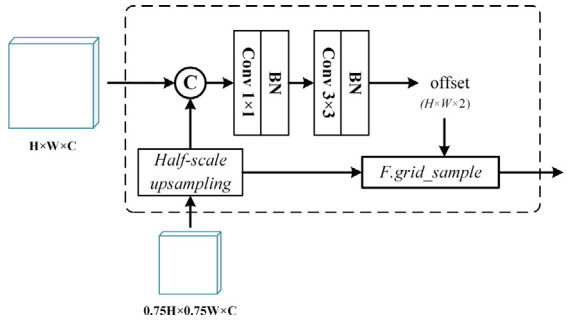

Fig. 4. The specific structure of half-scale alignment (HSA). The half-scale upsampling is to perform 1.5× upsampling on the deep feature, and then the upsampled feature and the local feature are aggregated and regressed to obtain the offset map of H× W× 2. Finally, the F.gridsample function is applied to perform pixel-level adjustments on the upsampled features to complete the process of feature alignment.

to generate some medium-scale features for transition, called transition layers. For example, between the features with scales of 1/8× (80 × 80 pixels), 1/16× (40 × 40 pixels), and 1/32× (20 × 20 pixels) relative to the input image I (640 × 640 pixels), we generate additional transition layers with scales of 60 × 60 and 30 × 30 pixels. This approach narrows the scale gap of features between different levels by half, and the obtained FPN structure is called Dense FPN. Note that the output of Dense FPN remains at the original three scales. To control the overall computational cost of the network, we use a C3 module with one residual block in the transition layers, and a C3 module with three residual blocks in the original layers.

The fusion operation between multi-scale features is an essential component of FPN-like methods. The medium-scale output of the Dense FPN is obtained by the fusion of two adjacent transition layers and local layers, which still has the problem of a large semantic gap between cross-scale features. Therefore, we further propose a novel progressive fusion method, which divides the fusion process of the three features into two stages. Different from ES-Net [32], we add the C3 module to fuse the features of each stage, which makes the feature transition gentler. In particular, before fusing the upsampled transition feature in the second stage, we perform a lightweight half-scale alignment (HSA) operation that adjusts the 1.5× upsampled transition feature at the pixel level to better match the local feature. By further processing the transition layer, the AD-FPN is obtained (see Fig. 3).

The structure of HSA is shown in Fig. 4, and its specific designs in our method are described in detail below. Similar to the works AlignSeg [18] and Semantic Flow [17], we use a lightweight convolutional subnetwork to regress the integrated feature map to obtain an offset heatmap of $H \times W \times 2$, and employ a parameter-free alignment function to further help align the upsampling results of the transition layers. It is worth noting that our method is more lightweight than FaPN [14], which also implements feature alignment in FPN. Our alignment function is based on the parameter-free grid sampling operation (function *F.grid_sample* in PyTorch), which can sample the feature map according to the learned offset, and does not require additional learnable parameters compared to DCN.

Unlike the spatial attention mechanism which requires a large receptive field to model long-range dependencies, the offset map pays more attention to the differences between pixels in the local region for finer adjustment. Therefore, we employ convolutions with small kernels (such as 3 × 3 and 1 × 1) in the process of regressing the offset map. Additionally, we change the partial upsampling operations in YOLOv5 from nearest neighbor upsampling to bilinear interpolation upsampling.

In our method, we only adjust upsampled features to align with local features, which can accelerate the learning process of the offset map. According to the value provided by the offset map as the coordinate information, we fill each pixel of the upsampled feature map on the output feature map to complete the alignment. The alignment process is formulated as follows.

$$\Delta^F = Conv_{3\times3}(Conv_{1\times1}(Concat(Upsample(F^D), F^L))), \quad (1)$$

$$F^A = Align(Upsample(F^D), \Delta^F), \quad (2)$$

where F^D and F^L represent the high-level feature and the local feature, respectively, and $Concat(x, y)$ denotes the concatenation

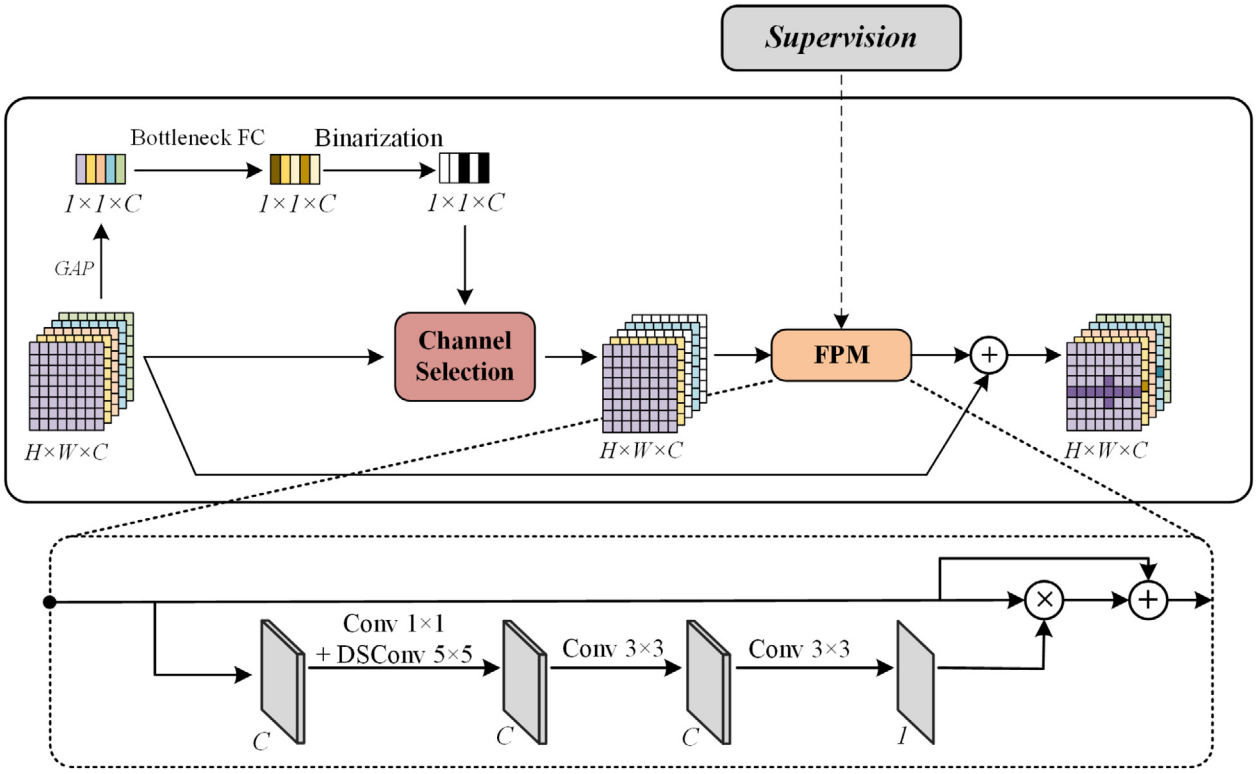

Fig. 5. The overall structure of AFPM. First, a SE-like structure is designed to dynamically select the position-sensitive channel Group G1. Then, in FPM, multi-stage convolution fusion is performed to obtain a spatial heatmap (mask), which is multiplied with G1 at the pixel level to achieve foreground enhancement and background weakening. Note that we impose strong supervision on the mask to help localization. Finally, the position-insensitive channels are restored through residual connection.

of x and y in the channel dimension. $Conv_{k\times k}$ is the convolution with $k \times k$ as the kernel size. Δ^F is the obtained offset map. F^A stands for the result of aligning the upsampled high-level feature (F^D) using the alignment function *Align*. Specifically, $Align(F, \Delta)$ indicates that a sample kernel is used to fill the output feature map with pixel values taken from the input feature map F according to the spatial location defined in Δ. It is defined as follows.

$$M_k^c = \sum_i^H \sum_j^W N_{ij}^c max(0, 1 - |x_k - i|) max(0, 1 - |y_k - j|), \tag{3}$$

where (x_k, y_k) represents the kth coordinate in the offset map Δ. N_{ij}^c denotes the pixel value at the (i, j) position on the cth channel of the input feature map, and H and W are the height and width of the input feature map, respectively. M_k^c stands for the pixel value of the kth pixel on the cth channel of the output feature map. The above formula indicates that the input feature map N is sampled according to each coordinate value in Δ, and then filled into the output feature map M. Note that the sample kernel we use is a bilinear interpolation kernel, so this process can realize back-propagation of gradients to achieve end-to-end training.

Extensive experimental results prove that AD-FPN has a great advantage in accuracy compared with other advanced FPN structures in the task of defect detection, while only bringing a small amount of extra computation.

3.3. Adaptive Feature Purification Mechanism (AFPM)

There are often large areas of obvious texture patterns in fabric images, and some non-salient defects are easily submerged in them. Note that the multi-scale features extracted by the backbone network cannot independently distinguish the foreground and background. Although there are FPN-like structures in the detection network to fuse multi-scale features, the lack of guidance from specific supervision signals still leads to unclear learning objectives and slow convergence.

In addition, after visualizing each channel under the same feature map, we find that many of these feature maps are not consistent with the intuitive visual features of humans. That is, some feature maps produce weaker or even no response in the defective area, possibly because the CNN describes the different aspects of the image in higher dimensions, not just from the perspective of location. To the best of our knowledge, most of the existing spatial attention methods [20,40] and some self-attention methods [42,44,45] generate a spatial heatmap of shape $(H \times W \times 1)$ to uniformly weight each pixel on the feature maps of all channels. However, since the feature maps mentioned above have the characteristics of diverse representations, such an approach will interfere with the features of many position-insensitive channels.

Therefore, we propose an adaptive feature purification mechanism, AFPM, to strengthen the features of those channels that are sensitive to defective regions and simultaneously filter out the interference of invalid features brought by the background, which is directly supervised as an independent branch. The structure of AFPM is shown in Fig. 5. The proposed AFPM is mainly divided into three steps: the first step is channel selection, the second step is the feature enhancement of defective regions and the filtering of background features, and the third step is feature recovery.

For an input feature of shape $H \times W \times C$, we first perform channel selection. The proposed channel selection mechanism is designed on the basis of simple SE attention, so subsequent work can be implemented with more efficient attention such as ECA. For the weight vector with shape $1 \times 1 \times C$ obtained after the sigmoid function, we binarize it with a threshold of 0.5 to obtain a binary vector. Using this binary vector as a criterion, we can distinguish some position-sensitive channels from features with multiple channels. In other words, we divide the channels into

two groups: one is position-sensitive (G1), and regional masks can be used on them to enhance the foreground features. The other group is position-insensitive (G2) and requires no additional processing. In particular, to keep the shape of the tensor consistent in subsequent fusions, we also keep G2 and only set its pixel value to 0. Then, we use a lightweight subnetwork to generate spatial masks, and a depthwise separable convolution (DSConv) with the large kernel to more efficiently aggregate global features. Moreover, to promote the convergence of network and strengthen the boundary of defects, we employ a strong supervision signal to supervise this subnetwork. The obtained mask is multiplied by G1, thus avoiding perturbation for position-insensitive channels. Finally, to recover the information of G2, we add the residual connection. Compared with the method used in FAN, we achieve better results with a significantly reduced number of parameters. The process of channel selection is formulated as follows.

$$z_1 = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} X(i,j), \tag{4}$$

$$z_2 = \sigma(W_2 ReLU(W_1 z_1)), \tag{5}$$

$$Gate[k] = \begin{cases} 0 & z_2[k] \leq 0.5 \\ 1 & z_2[k] > 0.5 \end{cases}, \tag{6}$$

$$X = Gate * X, \tag{7}$$

where X represents the input feature map, with width W and height H. W_1 and W_2 represent the weights of the FC layers, with shapes $\mathbb{R}^{\frac{C}{r} \times C}$ and $\mathbb{R}^{C \times \frac{C}{r}}$, respectively. The two FC layers constitute a bottleneck structure to reduce the number of parameters. σ represents the sigmoid function. The formula above describes such a process. First, the spatial information of the input feature X is aggregated to obtain z_1. Then the relationship between channels is modeled based on z_1 to obtain z_2. Binarization is performed based on z_2 to obtain the vector *Gate*. Finally, a channel-wise multiplication is performed between *Gate* and X to achieve channel selection for X.

The specific steps of FPM are described below. First, the processed C3 is passed through a 1×1 convolution and 5×5 DSConv (w/ SE) to further generate semantically rich features. Based on this integrated feature, we use a 3×3 convolution for regression to obtain the attention map. The obtained attention map is used to provide a mask for the low-level layer to filter out the interference of complex backgrounds and help the network to quickly focus on the defective regions. To avoid the information loss of the original feature caused by the mask and the vanishing/exploding problem of gradients, we use the residual connection.

Since YOLO adopts a multi-scale prediction paradigm, objects of different scales are independently predicted at different detection layers. Therefore, we set a scale threshold before generating the attention mask, which is used to assign the ground truth of objects at the corresponding scale to a specific branch so that the detection of objects at different scales does not interfere with each other. Furthermore, according to the structural design of the backbone in YOLOv5, we compute the upper bound of the receptive field at each prediction layer as the threshold. The calculation method is as follows, in which the effect of padding is ignored. Since we only generate a mask for the shallow layer here, the pixel area is set in the range of 1 to 5000.

$$RF_k = RF_{k-1} + (f_k - 1) * \prod_{i=1}^{k-1} s_i, \tag{8}$$

where the receptive field (RF) of the first layer is the kernel size of the convolutional layer, and k denotes the index of layer. RF_k represents the receptive field of the kth layer, and RF_{k-1} represents the receptive field of the (k-1)-th layer. f_k denotes the kernel size of the kth convolutional layer, and s_i represents the stride of the ith layer.

In particular, existing attention mechanisms are directly embedded in the network and then perform parameter updates as part of the network. Such an approach will lead to instability in the learning process due to the lack of a clear supervision signal for attention, further bringing negative effects to the network. Therefore, we employ a strong supervision signal to the mask attention subnet of AFPM so that the process of parameter updating can be guided directly according to the training labels, which can further release the learning burden of the network. Note that the pixel padding of the ground-truth box is used as the supervision signal for the FPM. The attention map generated by AFPM is similar to the result of semantic segmentation, so there is a large gap between the number of positive and negative samples (foreground and background pixels). Therefore, we use the combination of binary cross-entropy (BCE) loss and focal loss [46] as supervision, and update them together with the original loss function. It is not difficult to see that we are equivalent to adding a new branch to the YOLOv5 detection network and implementing a multi-task architecture. The overall loss function is shown in the following formula.

$$L_{mask} = \alpha L_{bce} + \beta L_{focal}, \tag{9}$$

$$L = L_{cls} + L_{bbox} + L_{obj} + L_{mask}, \tag{10}$$

where L_{cls}, L_{bbox}, and L_{obj} are the original classification, regression and confidence losses in YOLOv5, which will not be introduced here. L_{mask} is the loss we propose to supervise the update of mask, which is composed of BCE loss and focal loss, with the default initial value of 1 for the ratio of both.

3.4. Phase-wise feature redistribution module (PFRM)

When the feature pyramid network outputs multi-level features, the common practice of vanilla one-stage object detectors (such as YOLO) is to directly use a 1×1 convolution to adjust the number of channels, and then perform multi-scale prediction. However, the processing of multi-level output features leaves much room for improvement. Some novel solutions such as ASFF, ES-Net [32], Libra R-CNN [23], and YOLO-Ret [24] have been proposed.

During the processing of multi-level output features, we find that optimization difficulties arise when a single integrated feature is used to enhance multiple layers simultaneously. In addition, due to the limited representational ability of a single feature, there is an inability to provide effective guidance for multiple layers simultaneously, resulting in sub-optimal results, as in Libra R-CNN and YOLO-Ret. Different from the cascade fusion proposed by ES-Net, we innovatively design a phase-wise feature redistribution module to solve this problem, called PFRM, which generates features at different semantic levels containing global information. The structure of PFRM is shown in Fig. 6. Based on the obtained features at different semantic stages, we reassign features for each level, thus avoiding optimization difficulties and bringing further performance gains. The design process of the PFRM is described below.

First, the output features of three levels are concatenated in the channel dimension to obtain an integrated feature map, and then 1×1 convolution is used to reduce the subsequent computation. A lightweight channel attention module is employed to redistribute the weight of each channel, and the residual connection is attached here to preserve the original features. The channel attention we adopt is implemented based on ECANet.

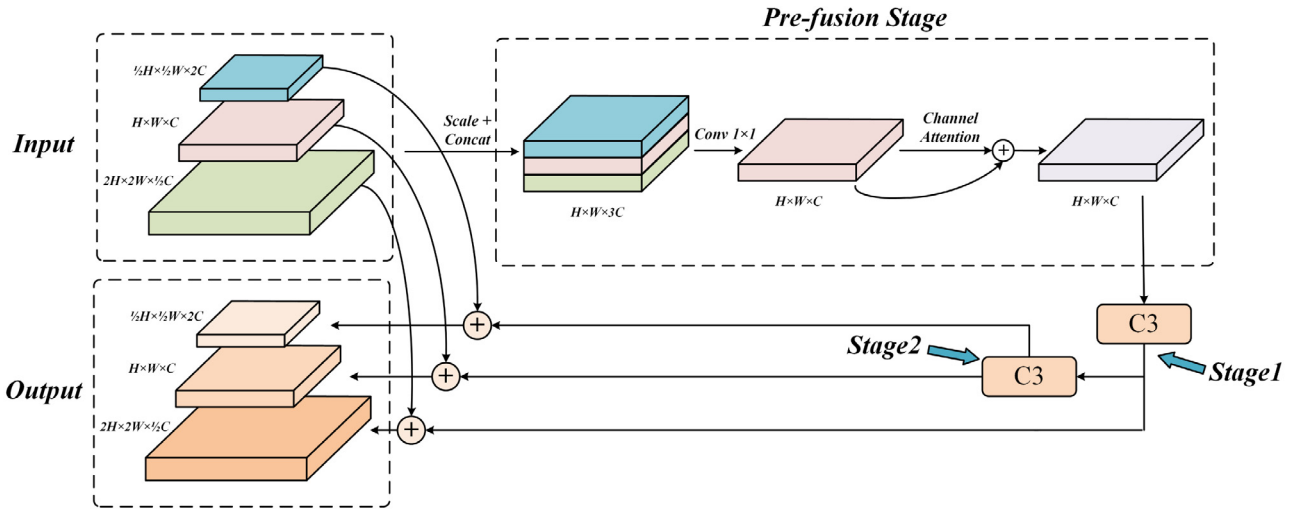

Fig. 6. The basic structure of the proposed PFRM. Note that the features at all levels of the output will be fused through the C3 module, which is not marked in the figure.

Then, based on the integrated features obtained in the pre-fusion stage, we generate features containing different semantic levels through several C3 modules, named Stage 1 and Stage 2. The feature Stage 1 with a lower semantic level is assigned to the low-level layer $M3$, and the feature Stage 2 with a higher semantic level is assigned to the two high-level layers $E4$ and $E5$. Through this approach, the global multi-level feature can be used to guide each layer in an adaptive manner, avoiding the difficulty of optimization and representation limitations of a single feature in previous methods.

Finally, for the features with different semantics generated by PFRM, we separately perform weighted summation with the features of the three original layers and use a lightweight C3 module for subsequent fusion at each layer.

Furthermore, by introducing PFRM before the prediction head, we strengthen the connection between features with a $2\times$ scale gap (as a supplement to half-scale feature fusion in AD-FPN), transitioning the dense scales of AD-FPN to the sparse scales required for final prediction (see [47]).

The pseudocode of the proposed CS-YOLO is provided in Algorithm 1.

Algorithm 1 Pseudocode of the Proposed CS-YOLO Framework

Input: RGB images with surface defects
Output: The predicted bounding boxes and corresponding classification results of defects

Use Mosaic, AugmentionHSV, RandomPerspective, Flip and Mixup techniques for data augmentation;
the Backbone part CSPDarknet_s returns feature maps $F = \{F3, F4, F5\}$;
the AD-FPN outputs feature maps $E = \{E3, E4, E5\}$;
the AFPM generates the mask K and transforms feature map $E3$ to $M3$;
the PFRM outputs feature maps $H = \{H3, H4, H5\}$;
for $i = 0; i < 3; i++$ **do**
    Perform regression and compression on feature $H(i+3)$;
    Obtain the predicted bounding box P_{bbox}, classification score P_{cls} and objectiveness confidence P_{obj};
    Calculate $Lhead_{bbox}(CIoU\ loss)$, $Lhead_{cls}(BCE\ loss)$ and $Lhead_{obj}(BCE\ loss)$ based on $\{P_{bbox}, P_{cls}, P_{obj}\}$; ▷ CIoU is proposed in [47]
    if i == 0 **then**
        Calculate L_{mask} based on K referring to Eq. (9);
    end if
    L_{bbox} += $Lhead_{bbox}$;
    L_{cls} += $Lhead_{cls}$;
    L_{obj} += $Lhead_{obj}$;
end for
Calculate total loss L of the Head part according to Eq. (10);
Apply NMS to the prediction boxes for refinement;
Get the final bounding boxes and corresponding classification scores.

4. Experimental results and analyses

We mainly verify the model performance and conduct ablation experiments based on the public Tianchi fabric dataset [48]. Subsequently, two representative public surface defect datasets, NEU-DET [49] and PCB defect dataset [50], are also used to test the comprehensive performance of the model and compare with some other SoTA methods. In addition, we conduct experiments on the remote sensing dataset RSOD [51] and Pothole Image Dataset (PID) [52] to verify the generalization ability of the model. Finally, we present the detection results of CS-YOLO on various datasets.

4.1. Dataset

4.1.1. Tianchi fabric dataset

The categories of the fabric defect dataset include 15 categories of defects such as stains, broken figures and watermarks. In our experiments, 9 categories of defects are selected as targets: sewing, sewing print, scrimp, bug, flaw, color shade, miss print, hole and fold. The original image is divided into two images of 2048×1696 pixels, and the defect-free images are removed. Overall, 4972 images are obtained to form a new dataset. Among them, 3592 images are randomly selected as the training set, 634 images as the validation set, and 746 images as the test set [32].

4.1.2. NEU-DET

NEU-DET is a dataset of 6 typical defects of hot rolled steel strip, including rolled-in scale (RS), patches (Pa), crazing (Cr), pitted surface (PS), inclusion (In) and scratches (Sc). There are 1,800 grayscale images in total, with 300 samples for each category of defects. The dataset is divided into training set and test set according to the ratio of 7:3.

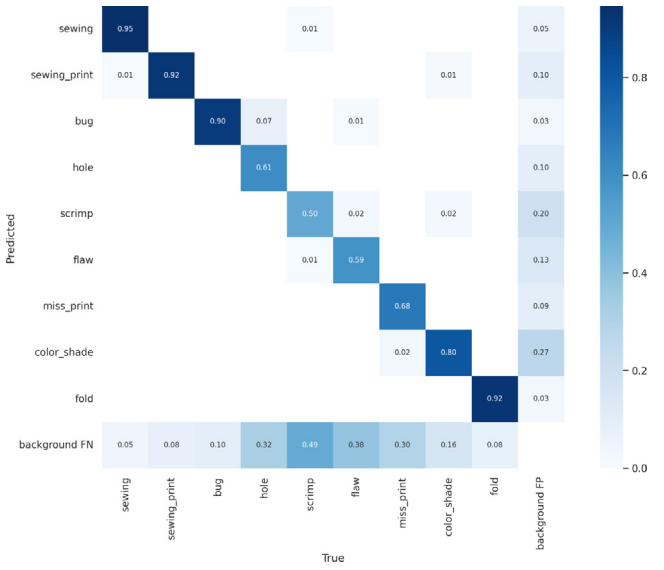

Fig. 7. Confusion matrix obtained by testing Tianchi fabric test set using YOLOv5s.

Table 1
The number of images and instances of each category in the RSOD dataset.

Category	Number/image	Number/instance
Aircraft	446	4993
Oil-tank	189	191
Playground	176	180
Overpass	165	1586

4.1.3. PCB defect dataset

PCB defect dataset is a public synthetic PCB dataset released by Peking University, which contains 1386 images and 6 kinds of defects (missing hole, mouse bite, open circuit, short, spur, spurious copper) for detection, classification and registration task.

4.1.4. RSOD

RSOD is an open dataset for object detection in remote sensing images. The dataset categories includes aircraft, oil tank, playground, and overpass, and the number of images and instances for each category is shown in Table 1. The dataset is divided into training set and test set in a ratio of 5:1.

4.1.5. Pothole image dataset (PID)

PID is a fully bounding-box annotated image dataset of potholes and damaged roads. It contains 700 images with 3k+ annotations on potholes and is divided into a training set and test set at a ratio of 8:2.

4.2. Implementation details

We use PyTorch to implement our method in PyCharm, and perform training and testing on NVIDIA RTX3060 GPU (with 12 GB memory). Since our backbone network is built based on YOLOv5s, we transfer some pretrained weights of YOLOv5s to our model during the training phase, which can save considerable training time. We use a stochastic gradient descent (SGD) optimizer for training, and the initial learning rate is set to 0.01 with a linear learning rate schedule. The weight decay of the optimizer is set to 0.0005 and the learning rate momentum is 0.937. We train the model for a total of 400 epochs and set the batch size to 16. In addition, we perform a learning rate warm-up at the beginning of training to ensure the stability of the model. Warm-up training is performed for 3 epochs, with an initial learning rate of 0.1 and a momentum of 0.8. Finally, since the data augmentation method is largely related to the characteristics of the images in the dataset, and different augmentation methods need to be used for each dataset, which is not within the scope of innovation in this paper. Therefore, in order to maintain the consistency with YOLOv5, the data augmentation method we use for each dataset is the initial mosaic data augmentation [53]. All images are scaled to 640 × 640 pixels before being input to the network for training.

Our evaluation metrics include the mean average precision (mAP), model storage size (Param), frames per second (FPS) and GFLOPs. Among them, mAP is the most important indicator to evaluate the accuracy of the object detection model, and Param is an indicator to measure the size of the model parameters, which is generally positively correlated with GFLOPs. FPS is an evaluation indicator to measure the efficiency of model inference, while Param and FPS are more important in resource-constrained scenarios.

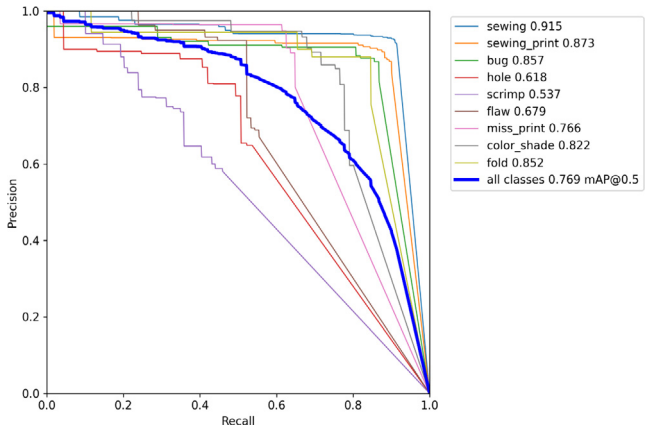

Fig. 8. The PR curve drawn based on the test results of YOLOv5s in the Tianchi fabric test set.

4.3. Preliminary analysis of YOLOv5s on the fabric dataset

By testing on the test set with YOLOv5s, we visualize the confusion matrix and Precision-Recall (PR) curve, as shown in Figs. 7 and 8, respectively. As seen from the confusion matrix, YOLOv5s performs well on most categories, and misclassification between defects in each categories is rare. The main problem is the confusion between background and defects, which is also a characteristic problem in the field of fabric defect detection. In addition, it can be observed from the PR curve that some hard categories such as flaw, hole, and scrimp have relatively poor results, and they are small to medium-sized defects that are susceptible to background interference.

4.4. Ablation study

4.4.1. Process analysis of implementing AD-FPN

To better understand the contribution of individual components to the model, we conduct ablation experiments. First, we gradually integrate various strategies used in AD-FPN and divide them into two categories, as shown in Table 2. G1 represents the basic architecture of Neck, and G2 represents the fusion method adopted. Note that a component with a tick after it indicates that it is selected, and Table 3 does the same. First, we test the performance of different architectures in Neck, such as PANet (Group A, default), BiFPN (Group C) and RevFP (Group D). Furthermore, Group B is obtained by generating more medium-scale layers in

Table 2
Progressive integration of AD-FPN and performance of each part.

		Methods	mAP/%	Param
G1	A	Vallina PANet (default)	76.9	14.5M
	B	A + Light transition layers (✓)	78.2	15.4M
	C	BiFPN [11]	76.6	15.4M
	D	RevFP [36]	75.8	16.0M
G2	E	B + Concat (default)	78.2	15.4M
	F	B + Weighted Add	76.6	14.6M
	G	E + Alignment	76.8	15.5M
	H	E + Cascade alignment fusion (✓)	78.9	16.6M

Table 3
Progressive integration of PFRM and performance of each part.

		Methods	mAP/%	Param
Kernel size of ECA	A	K = 3 (✓)	80.1	24.3M
	B	K = 5	79.0	24.3M
	C	K = 7	78.7	24.3M
Base module	D	C3 (n = 1) (✓)	80.1	24.3M
	E	C3 (n = 2)	80.1	24.4M
	F	DSConv5×5 (w/SE)	73.9	23.9M
Fusion operation	G	D + Add	79.2	24.3M
	H	D + Weighted add (✓)	80.1	24.3M
	I	D + Concat	80.1	25.0M

PANet. The results show that BiFPN and RevFP are lower than the default PANet in detection accuracy, so they cannot meet our expectations although they have fewer parameters. By introducing more medium-scale transition layers to PANet and using a light C3 module to help transition layers combine features, we obtain considerable accuracy gains at a small cost. This shows that the transition layer can indeed help the model to better integrate features with large differences in scales. Since transition layers are not used for prediction, such operations do not impose a large computational burden.

We further explore better fusion between adjacent layers based on Group B with the best results above. The comparison results between the three Groups E, F, and G show that the fusion methods of Weighted add or direct alignment in Concat achieved worse results. It can be concluded that the Concat operation itself can handle most fusion cases well, and we need to improve fusion nodes with greater potential for improvement. Therefore, in Group H, we adjust the three-input nodes existing in Group E, and divide the fusion process of three levels into two stages for transition in a gradual manner, alleviating the potential conflicts of multi-level features.

4.4.2. Process analysis of implementing PFRM

Based on the above AD-FPN, we further design PFRM to filter out noisy background information by utilizing the fusion results of multi-scale features. Similar to our progressive experiments with AD-FPN above, we finally achieve 80.1% mAP by fine-tuning the basic components.

First, we divide the internal components into three categories. The first category is the kernel size of ECA, the second category is the base module for staged fusion, and the third category is the operation of feature fusion. Unlike traditional attention such as SE, ECA achieves local correlation mapping between channels through fast 1D convolutions with kernel size k. Theoretically, the larger the k, the more stable the long-distance channel that can be constructed, and the better the performance. However, the experimental results show that the effect of k=3 is much better than that of k=5 and 7, which is even more surprising, indicating that the relationship between adjacent channels is more important. For the choice of the base module for staged feature fusion, we compare the effect of the C3 module with 1 and 2 residual blocks

Table 4
Progressive integration of AFPM and performance of each part.

Base Module	mAP/%	Param
–	80.1	24.3M
ASPP(multi-level)	79.4	26.8M
DSConv(multi-level)	79.1	25.6M
DSConv(single-level)	79.1	24.3M
DSConv(single-level, w/channel selection)	**80.8**	24.4M

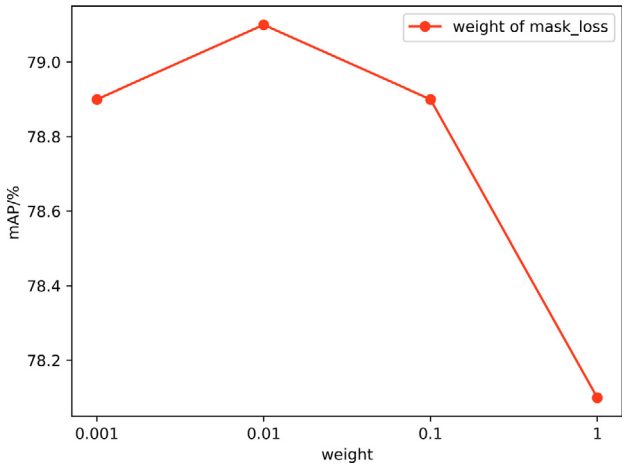

Fig. 9. Experiments with different weights of mask loss for supervised FPM.

and DSConv 5 × 5, respectively. From the experimental results of Groups D, E, and F, it can be seen that C3 is more suitable for iterative extraction of global information than DSConv, and the number of residual blocks has little effect on the feature extraction ability of C3. Finally, for the choice of fusion operations used in the phase-wise feature distribution, we experiment with Add, Weighted add, and Concat, respectively. It can be seen that there is no difference in accuracy between Concat and Weighted add, but Concat brings more parameters due to the increase in the number of channels. Therefore, we choose Weighted add as the fusion operation for feature redistribution.

4.4.3. Process analysis of implementing AFPM

After integrating the proposed AD-FPN and PFRM into YOLOv5s, we also conduct incremental experiments on AFPM, as shown in Table 4, where the adopted weight of mask loss is 0.01.

We first refer to the design patterns in FAN and SSPNet to aggregate features at three scales, and use ASPP to enhance the semantic representation of aggregated features. Through experimental comparison, it can be found that ASPP does perform better in accuracy, but it imposes many extra parameters and an inference burden. We then adopt a lightweight DSConv with a kernel size of 5 × 5. In particular, by using the proposed channel selection method for channel pre-grouping before FPM, we achieve a detection accuracy of 80.8% mAP based on DSConv. This result fully demonstrates that there is indeed a position-sensitive and position-insensitive distinction in multi-channel features, and that applying position masks to the features of position-insensitive channels actually leads to performance degradation. Moreover, we experiment with the weights of mask loss in the overall loss, as shown in Fig. 9.

Note that most of the above results are downgraded compared to the baseline (80.1% mAP). However, only our final AFPM achieves an improvement of 0.7% mAP with a negligible number of extra parameters. It is also worth noting that ordinary FPM can bring improvements in vanilla pyramid networks such as FPN or PAN, while for AD-FPN, a channel selection mechanism is

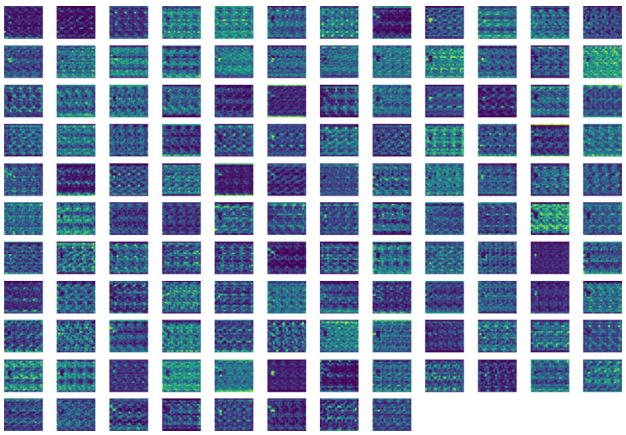

Fig. 10. Example of feature maps for individual channels of low-level features.

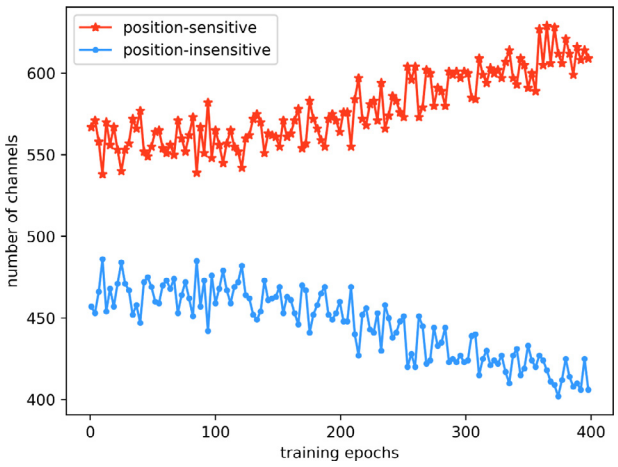

Fig. 11. Changes in the number of channels during training. The polyline formed by the red stars represents the number of channels with a weight of 0 (position-insensitive), and the polyline formed by the blue circles represents the number of channels with a weight of 1 (position-sensitive). To make the data points display more clearly, we calculate the number of channels every four epochs.

Table 5
Progressive integration of the proposed three components based on YOLOv5s.

AD-FPN	PFRM	AFPM	mAP/%	Param	GFLOPs
–	–	–	76.9	14.5M	15.9
✓	–	–	78.9	16.6M	18.6
✓	✓	–	80.1	24.3M	25.4
✓	✓	✓	80.8	24.4M	25.9

required to achieve the desired effect. This phenomenon reflects that after AD-FPN processing, the features increasingly show a trend of being position-independent, and can also be considered to contain a higher degree of semantics. This characteristic is more pronounced in higher-level layers of the network. Therefore, we visualize some low-level features in Fig. 10. It can be seen that although the network can localize defects well, the intensity of this response may appear opposite between channels (still maintaining high contrast). That is, directly applying a segmentation mask of defects to enhance defective regions may reduce the contrast of channel features, further burying the defects in the complex backgrounds.

To demonstrate this assumption, we plot the changes in the number of channels (both position-sensitive and position-insensitive) in the channel selection mechanism during training, as shown in Fig. 11. It can be seen that as the training process continues, more position-insensitive channels appear in the feature map. From another point of view, as the training progresses, the network extracts features with more semantic rather than texture details. Then our method can well avoid applying masks to this part of the channels, thus achieving better results.

4.4.4. Comprehensive performance of integrated model

After the progressive integration of three modules on YOLOv5s, we improve mAP from 76.9% to 80.8%, as can be seen from Table 5. Note that by replacing PANet with the designed AD-FPN, we increase mAP by 2% compared to YOLOv5s while only increasing the number of parameters by 1.1 MB. This fully illustrates the importance of solving the misalignment problem for existing FPN-based structures. Then, we further introduce PFRM to aggregate and redistribute the output features of AD-FPN, in order to better transition the dense scales to the sparse scales for subsequent predictions. It can be seen that although PFRM brings a relatively large (compared to YOLOv5s) number of additional parameters, the improvement brought is very considerable and stable (1.2% mAP), so we strongly encourage its introduction to the network. In particular, the cost of parameters is negligible in some large models. The improvement brought by PFRM shows that it is quite effective to supplement global information at different semantic levels for features at various scales. Finally, we introduce AFPM between AD-FPN and PFRM. By performing channel-wise enhancement in the shallow layer, we further improve the upper limit of the detector's accuracy to 80.8%. The result further demonstrates the importance of channel selection before performing regional enhancement.

Table 6
Comparison of the test results (mAP, Param and FPS) of various methods and proposed CS-YOLO on the Tianchi fabric dataset.

Methods	mAP@0.5	Param	FPS
Faster R-CNN [26]	60.8%	315.4M	16
Cascade R-CNN [27]	67.1%	552.9M	11
Sparse R-CNN [54]	66.5%	–	13
RetinaNet [46]	58.1%	246.5M	16
YOLOv3 [55]	56.44%	235.2M	37
YOLOv4 [53]	65.3%	244.2M	49
YOLOv4-CSP [56]	68.0%	200.4M	53
ES-Net [32]	76.2%	147.98M	56
YOLOv5 m [25]	76.9%	42.3M	73
YOLOv5l [25]	78.2%	92.9M	41
CS-YOLO(Ours)	**80.8%**	**24.4**M	**87**

4.5. Comparison with the state-of-the-art methods in various datasets

Tianchi fabric dataset, NEU-DET, PCB defect dataset, RSOD and PID are tested with the proposed method and the experimental results are compared with the benchmarks on the corresponding datasets.

First, we compare the proposed CS-YOLO with various state-of-the-art methods on the Tianchi fabric dataset, and some of the results are from [32], as shown in Table 6.

Without bells and whistles, our proposed CS-YOLO achieves the highest mAP value of 80.8% on the Tianchi fabric dataset, as shown in Table 6. Just by using the proposed three modules, we achieve a 4.1% mAP improvement over the baseline YOLOv5s. Since the baseline adopted is relatively lightweight, we are more concerned with improving accuracy as much as possible while ensuring high real-time performance. Compared with three two-stage methods, Faster R-CNN, Cascade R-CNN,

Table 7
Comparison of the test results (mAP) of various methods and proposed CS-YOLO on the RSOD.

Method	aircraft	oil-tank	overpass	playground	mAP (%)
R-P-Faster R-CNN [57]	70.8	90.2	78.7	98.1	84.5
Deformable R-FCN [58]	71.5	90.3	81.5	99.5	85.7
Deformable R-FCN and arcNMS [58]	71.9	90.4	89.6	99.9	87.9
Faster R-CNN [26] (VGG-16)	71.3	90.7	90.9	99.7	88.1
Faster R-CNN [26] (ResNet-101)	71.9	90.9	100	**100**	90.7
SE block [59] (VGG-16)	80.5	90.9	90.9	**100**	90.6
SE block [59] (ResNet-101)	77.4	90.9	**100**	99.1	91.8
RFN [59] (Faster R-CNN with VGG-16)	80.1	99.4	**100**	**100**	94.9
RFN [59] (Faster R-CNN with ResNet-101)	79.1	90.5	**100**	99.7	92.3
YOLOv3 [55]	84.8	99.1	81.2	**100**	91.27
M2Det [60] (VGG-16)	80.99	**99.98**	79.1	**100**	90.02
CF2PN [61] (VGG-16)	**95.52**	99.42	83.82	95.68	93.61
CS-YOLO(Ours)	93.9	98.7	93.4	98.9	**96.4**

Table 8
Comparison of the test results (mAP) of various methods and proposed CS-YOLO on the NEU-DET.

Methods	Backbone	mAP@0.5
SSD [62]	VGG16	71.6%
Faster R-CNN [26]	ResNet-34	70.0%
Faster R-CNN [26]	ResNet-50	76.6%
DEA-RetinaNet [63]	ResNet-50	79.1%
CABF-FCOS [64]	ResNet-50	76.7%
DDN [65]	VGG16	76.3%
DDN [65]	ResNet-50	**82.1%**
ES-Net [32]	CSPDarknet-53	79.1%
CS-YOLO(Ours)	CSPDarknet_s	78.2%

Table 9
Comparison of the test results (mAP) of various methods and proposed CS-YOLO on the PCB defect dataset.

Methods	Backbone	mAP@0.5
Faster R-CNN [26]	VGG-16	56.8%
Faster R-CNN [26]	ResNet-101	93.1%
FPN [9]	ResNet-101	92.7%
YOLOv4 [53]	CSPDarknet53	87.9%
Improved YOLOv4 [66]	CSPDarknet53	96.3%
ES-Net [32]	CSPDarknet53	97.5%
CS-YOLO(Ours)	CSPDarknet_s	**99.5%**

Table 10
Comparison of the test results (mAP) of various methods and proposed CS-YOLO on the PID.

Method	Backbone	mAP@0.5
SSD [62]	MobileNetv2	51.9%
SSD [62]	Lite MobileNetv2	55%
YOLOv3 [55]	DarkNet-53	66.3%
YOLOv3-SPP [55]	DarkNet-53	68.8%
YOLOv4 [53]	CSPDarkNet-53	77.7%
YOLOv4-tiny [53]	CSPDarkNet-53	78.7%
YOLOv5s [25]	CSPDarknet_s	74.8%
CS-YOLO(Ours)	CSPDarknet_s	**81.4%**

and Sparse R-CNN, our method not only improves significantly in accuracy, surpassing their mAP values of 20%, 13.7%, and 14.3%, respectively, but also has great advantages in model size and inference speed. Furthermore, for some one-stage methods, such as RetinaNet, YOLOv3, YOLOv4, and YOLOv4-CSP, we outperform them by 22.7%, 24.36%, 15.5% and 12.8% on mAP, respectively. Meanwhile, the parameter amount of CS-YOLO is only 12.5% of YOLOv4-CSP. Compared with the SoTA ES-Net method, we also surpass it in various indicators. As seen from the results of YOLOv5 m and YOLOv5l, scaling the network directly does not lead to reliable performance gains (at least in the field of defect detection). In contrast, our CS-YOLO further improves the network performance by solving some typical issues in YOLO and even general object detectors, and the efficiency cost is within an acceptable range.

Next, we train and test CS-YOLO on the NEU-DET, PCB defect dataset, RSOD and PID. The comparison results are compared with corresponding benchmarks on mAP metrics.

It can be seen from Table 7 that although CS-YOLO is not specifically optimized based on the dataset, our method also performs well on RSOD compared to some stronger benchmarks, such as RFN and CF2PN. In particular, it can be seen that most methods perform relatively poorly in the aircraft category. However, our CS-YOLO has a more balanced performance, thus reaching the state-of-the-art on the mAP metric, which also shows that CS-YOLO has better adaptability to class imbalance.

Likewise, our method achieves good performance on the public datasets NEU-DET, PCB dataset and PID (see Tables 8–10). Although we did not optimize the model for the dataset, we achieve results that are competitive with state-of-the-art detection methods. In particular, our method achieves a mAP of 99.5% on the PCB dataset, outperforming previous methods by a large margin. On NEU-DET, our results are comparable to ES-Net, reaching 78.2%. Note that DDN, DEA-RetinaNet and CABF-FCOS are specifically designed for the NEU-DET dataset. On the public kaggle competition dataset PID, we also achieve a detection accuracy of 81.3%. These results fully demonstrate the strong generalization ability of our method. We believe that after subsequent targeted optimization for specified scenarios, we can obtain better results. Finally, the strong capability of CS-YOLO once again demonstrates the effectiveness of the proposed three components (AD-FPN, PFRM and AFPM). Due to the versatility of the design, they can also be applied to other detectors to help improve performance with only a few adjustments. We briefly visualize the detection results using CS-YOLO on various datasets, as shown in Figs. 12–14. It can be seen that CS-YOLO has excellent detection results for targets in various complex scenarios.

4.6. Computational complexity

The space complexity can be represented by the number of model parameters, as shown in the Param column in Tables 2–6. As shown in Table 6, the number of parameters in our method is only 24.4M, and the amount of computation is only 25.9 GFLOPs. In terms of the number of parameter, CS-YOLO is nearly 1/10 of some vanilla one-stage methods such as RetinaNet (246.5M), YOLOv4 (244.2M), YOLOv4-CSP (200.4M), and 1/6 of the powerful ES-Net (147.98M). At the same time, it is also nearly 1/5 of the baseline network YOLOv5 series in the case of similar accuracy.

In addition, the inference speed can indicate the time complexity of the network. As shown in Table 6, we report the inference speed (i.e., FPS) of various methods under the same environment. It can be seen that the inference speed of our method (87 FPS) is also significantly higher than that of other

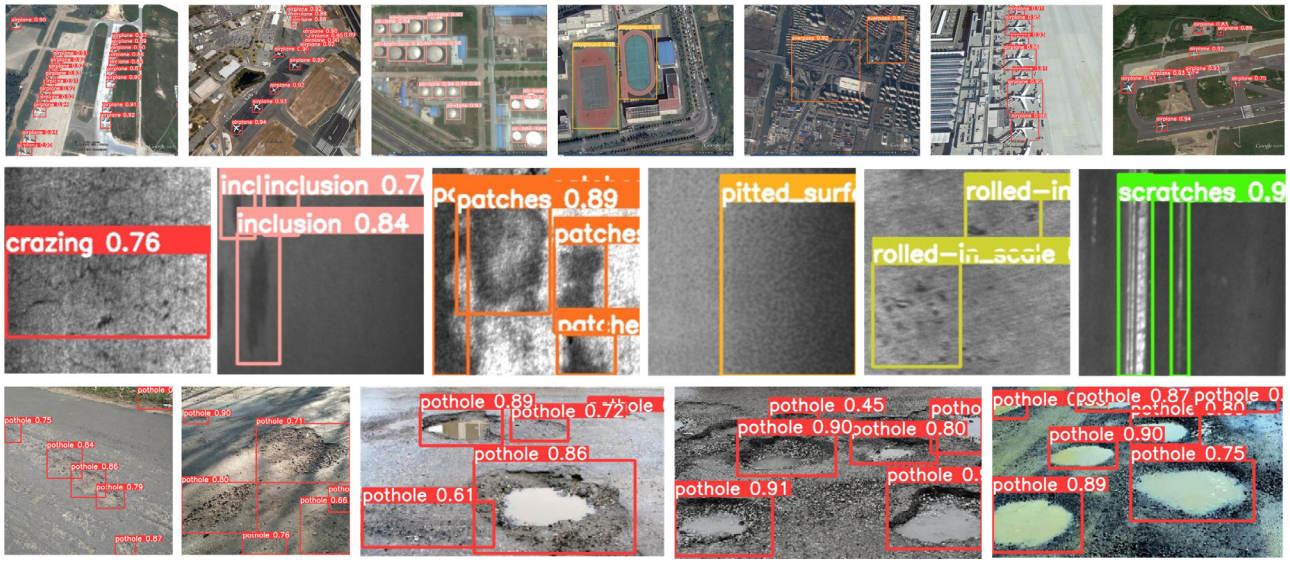

Fig. 12. The detection results on the RSOD, NEU-DET, and PID datasets are briefly summarized, corresponding to the first, second, and third rows, respectively.

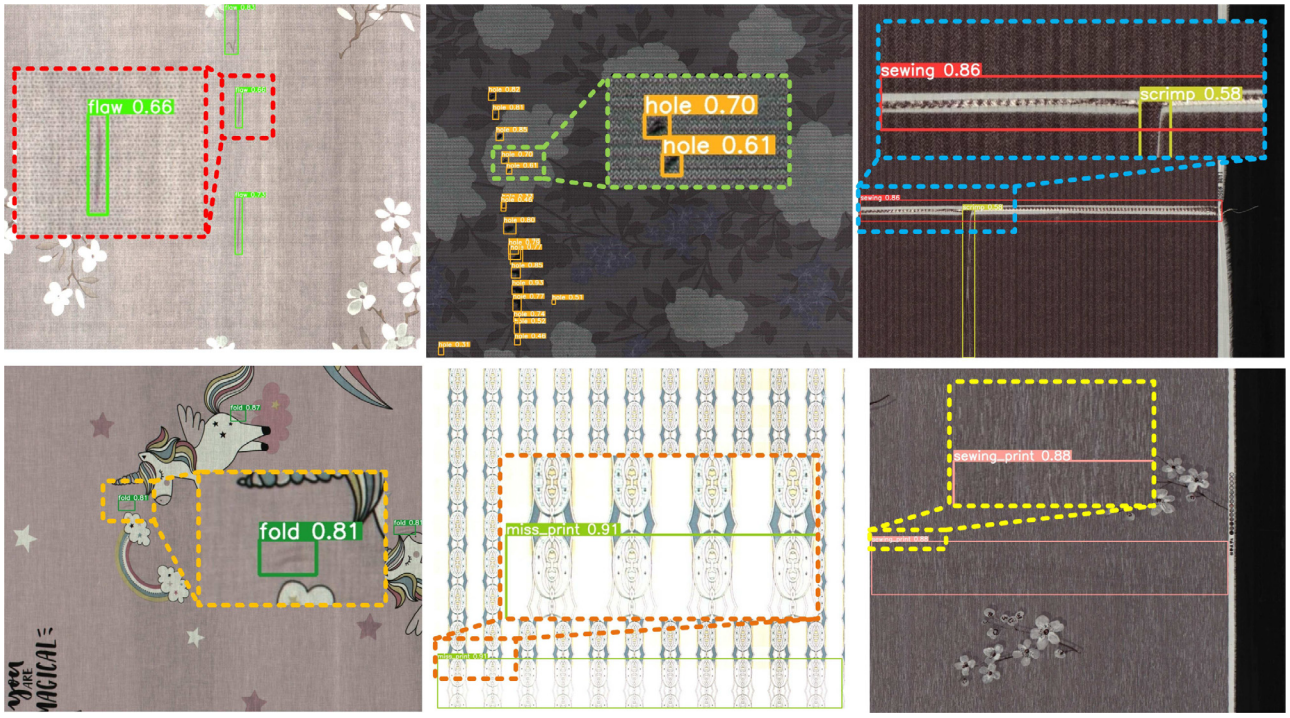

Fig. 13. Some representative detection results for the Tianchi fabric dataset are listed. Due to the high resolution of the image, we zoom in on the regions of the defects.

SoTA methods, while maintaining better accuracy performance, which proves that our method achieves a better trade-off in time complexity.

It is worth mentioning that YOLOv5 m and YOLOv5l are obtained by linear scaling compared to our baseline YOLOv5s. Although they increase the computational complexity and time complexity of the model, they fail to achieve a corresponding improvement in accuracy. In particular, the accuracy of YOLOv5 m is equal to that of YOLOv5s. This shows that simply increasing the complexity (model capacity) does not bring ideal gains in defect detection tasks, which also implies the need for more refined strategies to help detection in complex scenarios.

5. Conclusion

In the process of industrial production, surface defect detection has received increasing attention as a quality inspection method. Due to the particularity of defects, detailed information plays a particularly critical role in defect classification and localization. We believe that the following issues are of great significance in improving the performance of general object detectors in the task of defect detection. (1) The $2\times$ upsampling operation in FPN-based network will confuse object details (feature misalignment) due to pixel-level deviation between features at different scales. (2) The lack of direct interaction between cross-scale

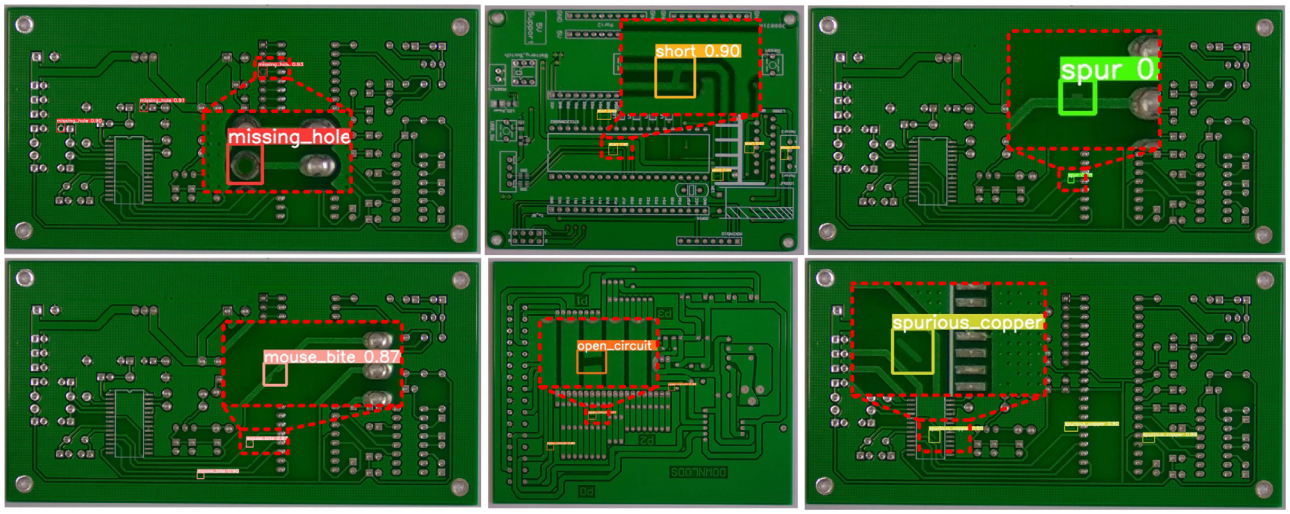

Fig. 14. Some representative detection results for the PCB dataset are listed. Like the Tianchi dataset, we zoomed in on the regions of the defects.

features leads to insufficient utilization of global information for the entire feature pyramid. (3) In addition, the complex backgrounds have serious interference with some non-salient defects. Therefore, to address these issues, this paper correspondingly proposes three novel components (AD-FPN, PFRM and AFPM) to form a progressive redistribution pyramid network CS-YOLO that can be applied to defect detection in complex scenarios. Note that the proposed method follows the design principles of the baseline network, so it can also be extended to accommodate scenarios with different resource constraints. Extensive experimental results on multiple datasets show that our method outperforms other state-of-the-art methods in most metrics. In the future, we will apply TensorRT for optimization, which can accelerate CS-YOLO to have the best real-time performance on edge devices (such as NVIDIA Jetson AGX Xavier) in complex industrial scenarios while retaining high detection accuracy.

CRediT authorship contribution statement

Xuyi Yu: Conceptualization, Methodology, Software, Writing – original draft. **Wentao Lyu:** Data curation, Writing – original draft, Investigation. **Chengqun Wang:** Visualization, Investigation. **Qing Guo:** Software, Validation. **Di Zhou:** Data curation, Writing – reviewing and editing. **Weiqiang Xu:** Supervision, Resources.

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.

Data availability

Data will be made available on request.

Acknowledgments

The work is supported by the National Natural Science Foundation of China (No. 61601410), the State Key Laboratory of Synthetical Automation for Process Industries, China (No. 2021-KF-21-03, 2021-KF-21-06), the Key Research and Development Program Foundation of Zhejiang (No. 2021C01047, 2022C01079).

References

[1] L. Yang, J. Fan, B. Huo, E. Li, Y. Liu, A nondestructive automatic defect detection method with pixelwise segmentation, Knowl.-Based Syst. 242 (2022) 108338, http://dx.doi.org/10.1016/j.knosys.2022.108338.
[2] D. Lin, Y. Li, S. Prasad, T.L. Nwe, S. Dong, Z.M. Oo, CAM-guided Multi-Path Decoding U-Net with Triplet Feature Regularization for defect detection and segmentation, Knowl.-Based Syst. 228 (2021) 107272.
[3] M.M. Ferdaus, B. Zhou, J.W. Yoon, K.L. Low, J. Pan, J. Ghosh, M. Wu, X. Li, A.V.-Y. Thean, J. Senthilnath, Significance of activation functions in developing an online classifier for semiconductor defect detection, Knowl.-Based Syst. 248 (2022) 108818.
[4] X. Li, X. Yang, X. Li, S. Lu, Y. Ye, Y. Ban, GCDB-UNet: A novel robust cloud detection approach for remote sensing images, Knowl.-Based Syst. 238 (2022) 107890, http://dx.doi.org/10.1016/j.knosys.2021.107890.
[5] Y. Chang, J. Chen, Q. Chen, S. Liu, Z. Zhou, CFs-focused intelligent diagnosis scheme via alternative kernels networks with soft squeeze-and-excitation attention for fast-precise fault detection under slow & sharp speed variations, Knowl.-Based Syst. 239 (2022) 108026.
[6] C. Tao, H. He, F. Xu, J. Cao, Stereo priori RCNN based car detection on point level for autonomous driving, Knowl.-Based Syst. 229 (2021) 107346, http://dx.doi.org/10.1016/j.knosys.2021.107346.
[7] Z. Zeng, B. Liu, J. Fu, H. Chao, Reference-based defect detection network, IEEE Trans. Image Process. 30 (2021) 6637–6647.
[8] H. Fang, M. Xia, H. Liu, Y. Chang, L. Wang, X. Liu, Automatic zipper tape defect detection using two-stage multi-scale convolutional networks, Neurocomputing 422 (2021) 34–50.
[9] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, S. Belongie, Feature pyramid networks for object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 2117–2125.
[10] S. Liu, L. Qi, H. Qin, J. Shi, J. Jia, Path aggregation network for instance segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 8759–8768.
[11] M. Tan, R. Pang, Q.V. Le, Efficientdet: Scalable and efficient object detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 10781–10790.
[12] S. Zhang, L. Wen, X. Bian, Z. Lei, S.Z. Li, Single-shot refinement neural network for object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4203–4212.
[13] Y. Chen, C. Han, N. Wang, Z. Zhang, Revisiting feature alignment for one-stage object detection, 2019, arXiv preprint arXiv:1908.01570.
[14] S. Huang, Z. Lu, R. Cheng, C. He, FaPN: Feature-aligned pyramid network for dense image prediction, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 864–873.
[15] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, Y. Wei, Deformable convolutional networks, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 764–773.
[16] M. Jaderberg, K. Simonyan, A. Zisserman, et al., Spatial transformer networks, Adv. Neural Inf. Process. Syst. 28 (2015).
[17] X. Li, A. You, Z. Zhu, H. Zhao, M. Yang, K. Yang, S. Tan, Y. Tong, Semantic flow for fast and accurate scene parsing, in: European Conference on Computer Vision, Springer, 2020, pp. 775–793.

[18] Z. Huang, Y. Wei, X. Wang, W. Liu, T.S. Huang, H. Shi, Alignseg: Feature-aligned segmentation networks, IEEE Trans. Pattern Anal. Mach. Intell. 44 (1) (2021) 550–557.
[19] J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7132–7141.
[20] 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.
[21] J. Wang, Y. Yuan, G. Yu, Face attention network: An effective face detector for the occluded faces, 2017, arXiv preprint arXiv:1711.07246.
[22] M. Hong, S. Li, Y. Yang, F. Zhu, Q. Zhao, L. Lu, SSPNet: Scale selection pyramid network for tiny person detection from UAV images, IEEE Geosci. Remote Sens. Lett. 19 (2021) 1–5.
[23] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, D. Lin, Libra r-cnn: Towards balanced learning for object detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 821–830.
[24] P. Ganesh, Y. Chen, Y. Yang, D. Chen, M. Winslett, YOLO-ReT: Towards high accuracy real-time object detection on edge GPUs, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2022, pp. 3267–3277.
[25] G. Jocher, et al., yolov5, https://github.com/ultralytics/yolov5.
[26] 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) (2016) 1137–1149.
[27] Z. Cai, N. Vasconcelos, Cascade r-cnn: Delving into high quality object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 6154–6162.
[28] D. Li, Q. Xie, Z. Yu, Q. Wu, J. Zhou, J. Wang, Sewer pipe defect detection via deep learning with local and global feature fusion, Autom. Constr. 129 (2021) 103823.
[29] Y. Wu, Z. Yi, Automated detection of kidney abnormalities using multi-feature fusion convolutional neural networks, Knowl.-Based Syst. 200 (2020) 105873.
[30] B. Su, H. Chen, Z. Zhou, BAF-detector: An efficient CNN-based detector for photovoltaic cell defect detection, IEEE Trans. Ind. Electron. 69 (3) (2021) 3161–3171.
[31] N. Zeng, P. Wu, Z. Wang, H. Li, W. Liu, X. Liu, A small-sized object detection oriented multi-scale feature fusion approach with application to defect detection, IEEE Trans. Instrum. Meas. 71 (2022) 1–14, http://dx.doi.org/10.1109/TIM.2022.3153997.
[32] X. Yu, W. Lyu, D. Zhou, C. Wang, W. Xu, ES-Net: Efficient scale-aware network for tiny defect detection, IEEE Trans. Instrum. Meas. 71 (2022) 1–14, http://dx.doi.org/10.1109/TIM.2022.3168897.
[33] Z. Yang, S. Liu, H. Hu, L. Wang, S. Lin, Reppoints: Point set representation for object detection, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 9657–9666.
[34] J. Wang, K. Chen, S. Yang, C.C. Loy, D. Lin, Region proposal by guided anchoring, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 2965–2974.
[35] G. Ghiasi, T.-Y. Lin, Q.V. Le, Nas-fpn: Learning scalable feature pyramid architecture for object detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 7036–7045.
[36] Z. Zong, Q. Cao, B. Leng, Rcnet: Reverse feature pyramid and cross-scale shift network for object detection, in: Proceedings of the 29th ACM International Conference on Multimedia, 2021, pp. 5637–5645.
[37] S. Liu, D. Huang, Y. Wang, Learning spatial fusion for single-shot object detection, 2019, arXiv preprint arXiv:1911.09516.
[38] Y.-M. Zhang, C.-C. Lee, J.-W. Hsieh, K.-C. Fan, CSL-YOLO: A new lightweight object detection system for edge computing, 2021, arXiv preprint arXiv:2107.04829.
[39] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, Q. Hu, ECA-Net: efficient channel attention for deep convolutional neural networks, 2020 IEEE, in: CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2020.
[40] J. Park, S. Woo, J.-Y. Lee, I.S. Kweon, Bam: Bottleneck attention module, 2018, arXiv preprint arXiv:1807.06514.
[41] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, H. Lu, Dual attention network for scene segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 3146–3154.
[42] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).
[43] K. He, X. Zhang, S. Ren, J. Sun, Identity mappings in deep residual networks, in: European Conference on Computer Vision, Springer, 2016, pp. 630–645.
[44] X. Wang, R. Girshick, A. Gupta, K. He, Non-local neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7794–7803.
[45] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin transformer: Hierarchical vision transformer using shifted windows, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 10012–10022.
[46] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2980–2988.
[47] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, D. Ren, Distance-IoU loss: Faster and better learning for bounding box regression, in: Proceedings of the AAAI Conference on Artificial Intelligence, 34, (07) 2020, pp. 12993–13000.
[48] A. Tianchi, Smart Diagnosis of Cloth Flaw Dataset, https://tianchi.aliyun.com/dataset/dataDetail?dataId=79336.
[49] Y.Y. Kechen Song, NEU surface defect database, http://faculty.neu.edu.cn/songkechen/zh_CN/zdylm/263270/list/index.htm.
[50] P. Wei, Public synthetic PCB dataset, http://robotics.pkusz.edu.cn/resources/dataset/.
[51] Y. Long, Y. Gong, Z. Xiao, Q. Liu, Accurate object localization in remote sensing images based on convolutional neural networks, IEEE Trans. Geosci. Remote Sens. 55 (5) (2017) 2486–2498.
[52] A. Rahman, S. Patel, Annotated potholes image dataset, 2020.
[53] A. Bochkovskiy, C.-Y. Wang, H.-Y.M. Liao, Yolov4: Optimal speed and accuracy of object detection, 2020, arXiv preprint arXiv:2004.10934.
[54] P. Sun, R. Zhang, Y. Jiang, T. Kong, C. Xu, W. Zhan, M. Tomizuka, L. Li, Z. Yuan, C. Wang, et al., Sparse r-cnn: End-to-end object detection with learnable proposals, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 14454–14463.
[55] J. Redmon, A. Farhadi, Yolov3: An incremental improvement, 2018, arXiv preprint arXiv:1804.02767.
[56] C.-Y. Wang, A. Bochkovskiy, H.-Y.M. Liao, Scaled-yolov4: Scaling cross stage partial network, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 13029–13038.
[57] X. Han, Y. Zhong, L. Zhang, An efficient and robust integrated geospatial object detection framework for high spatial resolution remote sensing imagery, Remote Sens. 9 (7) (2017) 666.
[58] Z. Xu, X. Xu, L. Wang, R. Yang, F. Pu, Deformable convnet with aspect ratio constrained nms for object detection in remote sensing imagery, Remote Sens. 9 (12) (2017) 1312.
[59] K. Zhou, Z. Zhang, C. Gao, J. Liu, Rotated feature network for multiorientation object detection of remote-sensing images, IEEE Geosci. Remote Sens. Lett. 18 (1) (2021) 33–37, http://dx.doi.org/10.1109/LGRS.2020.2965629.
[60] Q. Zhao, T. Sheng, Y. Wang, Z. Tang, Y. Chen, L. Cai, H. Ling, M2det: A single-shot object detector based on multi-level feature pyramid network, in: Proceedings of the AAAI Conference on Artificial Intelligence, 33, (01) 2019, pp. 9259–9266.
[61] W. Huang, G. Li, Q. Chen, M. Ju, J. Qu, CF2PN: A cross-scale feature fusion pyramid network based remote sensing target detection, Remote Sens. 13 (5) (2021) 847.
[62] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A.C. Berg, Ssd: Single shot multibox detector, in: European Conference on Computer Vision, Springer, 2016, pp. 21–37.
[63] X. Cheng, J. Yu, RetinaNet with difference channel attention and adaptively spatial feature fusion for steel surface defect detection, IEEE Trans. Instrum. Meas. 70 (2020) 1–11.
[64] J. Yu, X. Cheng, Q. Li, Surface defect detection of steel strips based on anchor-free network with channel attention and bidirectional feature fusion, IEEE Trans. Instrum. Meas. (2021).
[65] Y. He, K. Song, Q. Meng, Y. Yan, An end-to-end steel surface defect detection approach via fusing multiple hierarchical features, IEEE Trans. Instrum. Meas. 69 (4) (2019) 1493–1504.
[66] H. Xin, Z. Chen, B. Wang, PCB electronic component defect detection method based on improved YOLOv4 algorithm, in: Journal of Physics: Conference Series, 1827, (1) IOP Publishing, 2021, 012167.