# Improved YOLOv8-Based Defect Detection Model for Hot Rolled Strip Steel

Bingtian Qiao *

Posted Date: 8 April 2025

doi: 10.20944/preprints202503.1303.v2

Keywords: Hot-rolled strip steel; YOLOv8; attention mechanism; defect detection; lightweight convolution

 doi:10.20944/preprints202503.1303.v2



*Article*

# Improved YOLOv8-Based Defect Detection Model for Hot Rolled Strip Steel

**Bingtian Qiao**

Robotics and Intelligent Devices, Fuzhou University, Fuzhou, Fujian Province, 350100, China;
BINGTIAN.QIAO.2024@mumail.ie

**Abstract:** Production defects in hot-rolled strip steel, arising from process design issues or variations in material properties, adversely affect both economic returns and production safety. Existing deep learning–based surface defect detection methods are often too slow and computationally heavy For time-critical industrial operations. This paper proposes an optimized YOLOv8s algorithm for defect detection on hot-rolled steel strips. By integrating Cloformer with the CBAM attention mechanism, the model effectively enhances its capability to capture spatial relationship features. Additionally, the combination of partial convolution and depthwise separable convolution significantly reduces computational load. The introduction of the SimSPPF module further improves inference speed. Experimental experimental data reveals that the improved YOLOv8s model achieves 80.4% mAP (mean average precision) and 182.4 FPS inference speed while substantially reducing model size compared to conventional approaches.

**Keywords:** hot-rolled strip steel; YOLOv8; attention mechanism; defect detection; lightweight convolution

## 1. Introduction

Thanks to its excellent plasticity and ease of forming, hot-rolled strip steel is not only essential for producing cold-rolled plates but also plays a critical role in fabricating large-scale structural elements, forgings, and automotive components [1]. Nonetheless, limitations in production machinery and processes inevitably lead to surface imperfections. As illustrated in Figure 1, the primary defects include inclusions, crazing, rolled-in marks, patches, pitted areas, and scratches. These flaws not only detract from the product's appearance but, when the steel is used as a structural material, can act as stress concentrators that undermine performance and pose significant safety hazards [2]. Consequently, developing precise and efficient defect detection methods proves crucial for manufacturing enterprises to implement rigorous quality control and enhance overall safety standards.

With the rise of the Internet and the vigorous development of deep learning technology, more and more machine vision tasks adopt deep learning models. Relative to established machine vision methodologies, deep learning-based machine vision defect detection methods typically exhibit superior detection accuracy, greater adaptability and a wider range of applications.

 doi:10.20944/preprints202503.1303.v2



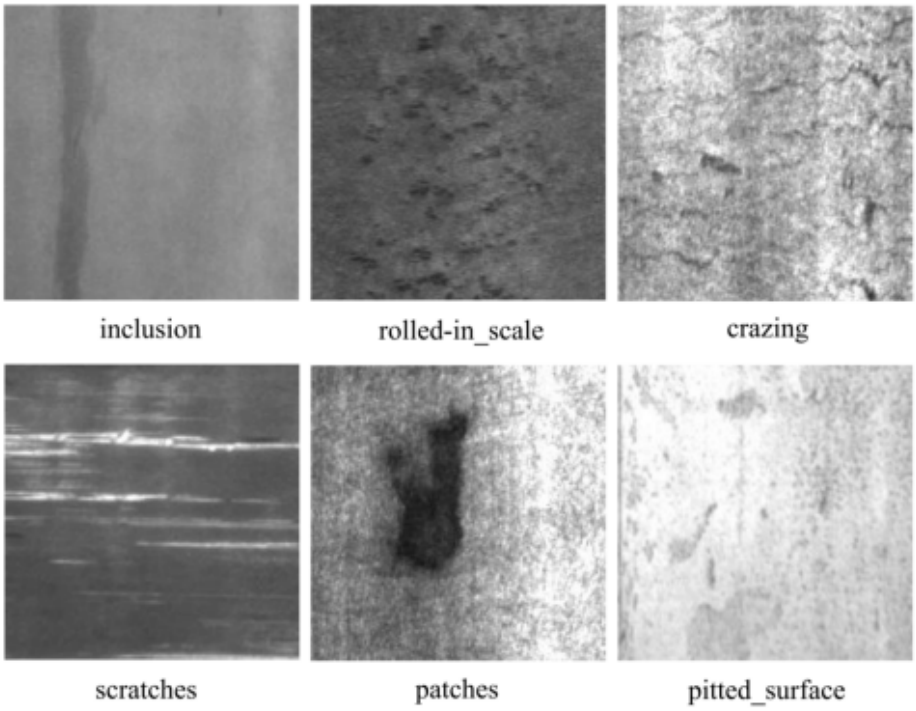


**Figure 1.** Six types of surface defects of hot-rolled strip steel.

The YOLO series represents a prominent deep learning object detection approach. Initially proposed by Joseph et al. in 2015, YOLOv1 [3] pioneered single-stage detection algorithms by significantly improving detection speed while maintaining reasonable accuracy. As a representative single-stage method, YOLO directly generates the existence probability and position coordinates of the de-tected objects during the detection process. Its characteristic is a relatively fast speed but at the expense of detection accuracy. YOLO algorithms directly predict target existence probability and positional coordinates during detection, offering rapid detection speeds at the expense of some precision reduction.

In contrast, the two-stage method illustrated by the Faster Region-based Convolutional Neural Networks (Faster R-CNN) [4] first extracts candidate boxes and then outputs the detection results. It has an advantage in detection accuracy but has certain limitations in real-time performance. To address the accuracy-speed trade-off in defect detection, Feng et al. [5] developed an enhanced RepVGG architecture integrated with spatial attention mechanisms. Building upon these developments, Shun et al. [6] demonstrated superior performance over YOLOv4 through a YOLOv5-based framework optimized for hot-rolled steel strip surface defects. Further advancing feature extraction capabilities, researchers [7] incorporated the BI-SPPFCSPC structure into YOLOv7's feature pyramid, significantly improving small object detection precision. Zhang et al. [8] introduced the Content-Aware Reassembly of Features (CARE) upsampling operator into the YOLOv8s model, effectively enhancing the feature extraction capability. Wang et al. [9] introduced ShuffleNetv2 as the backbone network in YOLOv5s and, through depthwise separable convolution and channel shuffling techniques, This approach substantially reduces model parameters while maintaining high detection accuracy. The above studies continuously explore a reasonable balance between model lightweight and detection accuracy, effectively improving detection accuracy and reducing inference time. However, they have not achieved a perfect balance among model parameters, computational cost, model size, computational accuracy, and inference time, and thus cannot fully meet the requirements of real-time hot-rolled steel strip defect detection. Given the stringent real-time and edge deployment demands in industrial settings [10], making the model more lightweight is extremely important.

 doi:10.20944/preprints202503.1303.v2



To meet stringent real-time detection requirements concerning surface faults in hot-rolled steel strips, this study optimizes the YOLOv8s network through three aspects: accuracy enhancement, model complexity reduction, and inference acceleration. First, by integrating Cloformer [11] with CBAM [12], the approach maximizes detection accuracy without a significant increase in model parameters. Second, the incorporation of depthwise separable convolution (DWconv) [13] alongside partial convolution (Pconv) [14] refines the backbone network and C2f module, substantially reducing both model parameters and computational complexity. Finally, SimSPPF [15] is employed to effectively extract and merge multi-scale features, these modifications not only improve multi-scale target recognition but also accelerate inference speed. Together, these improvements enable the model to be deployed on-site, meeting the stringent speed and accuracy requirements necessary for real-time monitoring.

## 2. YOLOv8 Overview

The YOLO family of algorithms is renowned for its ability to perform real-time object detection while learning features effectively and generalizing well across complete images. Leveraging improvements from YOLOv5, the latest version, YOLOv8, achieves cutting-edge performance in object detection tasks [16]. Based on network depth and width, YOLOv8 offers five model variants: YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x. For optimal balance between accuracy, speed, and model size in hot-rolled steel strip defect detection, YOLOv8s serves as the starting point for optimization. The YOLOv8s architecture is primarily arranged into four components: the input module, Backbone, Neck, and Head, as illustrated in Figure 2.

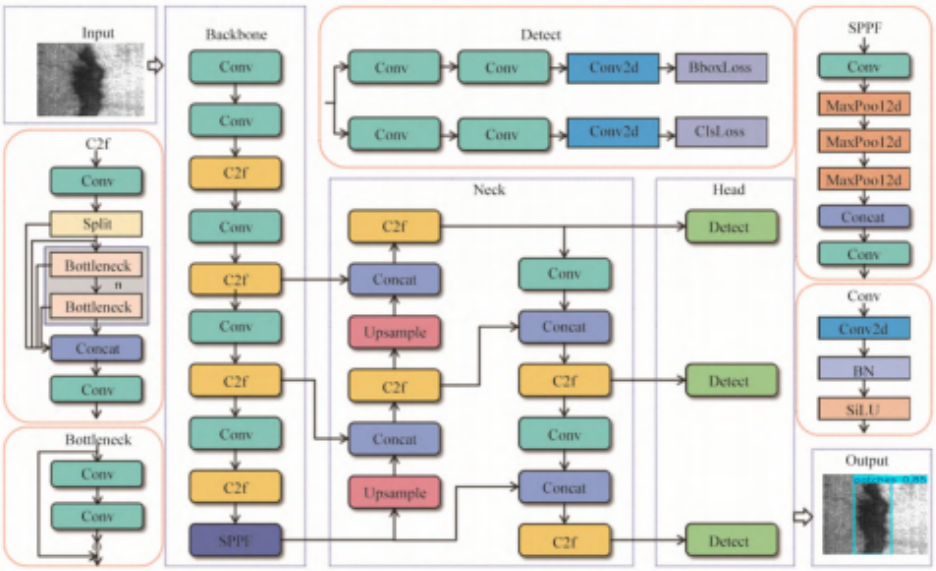


**Figure 2.** YOLOv8s Network Structure Diagram [17].

### 2.1. Input End

The input of YOLOv8 is standardized images, supporting multi-scale input (such as 640×640, 1280×1280, etc.). The preprocessing includes image normalization, data augmentation, and adaptive scaling. Image normalization is responsible for normalizing pixel values from [0, 255] to [0, 1]. Data augmentation adopts Mosaic data augmentation (random cropping, rotation, flipping, etc.), and Mosaic is turned off in the later stage of training to improve accuracy and enhance convergence [18]. Adaptive scaling maintains the aspect ratio of the image through the Letterbox method and fills gray borders to avoid distortion.

 doi:10.20944/preprints202503.1303.v2



### 2.2. Backbone

The backbone module primarily handles feature extraction through three core components: the C2f module [19] (cross-stage fusion with dual convolution and merging), SPPF module (fast spatial pyramid pooling), and fundamental convolutional blocks.

#### 2.2.1. C2f

C2f replaces the C3 module in YOLOv5 with a design inspired by the ELAN structure in YOLOv7, incorporating multiple Bottleneck modules and cross-layer connections. Initial channel compression is achieved through 1×1 convolution., then splitting the output into two branches. One branch is processed through several Bottleneck modules, while the other is left unchanged. After merging these two branches, a 1×1 convolution is applied to restore the channel dimensions. Compared to C3, the C2f module reduces The number of variables by about 20% while strengthening gradient flow and feature reuse.

#### 2.2.2. SPPF

The SPPF module achieves multi-scale feature fusion by concatenating multiple 5×5 max pooling layers in series. The input passes through three pooling layers (stride = 1, padding = 2) in sequence, and the results are concatenated and compressed through convolution. The proposed mechanism enhances spatial awareness through expanded receptive fields, enabling superior generalization to objects at disparate scales.

### 2.3. Neck

The Neck module employs an improved PAN-FPN structure for cross-scale feature combinatio. In this design, the FPN pathway [20] conveys nuanced semantic patterns from smaller feature maps downwards, while the PAN [21] pathway transmits low-level location details from larger feature maps upwards. The process involves first aligning feature map sizes via upsampling (for example, through nearest neighbor interpolation), then concatenating features from various Backbone stages, and finally applying the C2f module to further integrate the fused features.

### 2.4. Head

Head is responsible for object detection and classification, adopting a decoupled head and anchor-free mechanism.

#### 2.4.1. Decoupled Head

The decoupled detection head utilizes binary cross-entropy loss (BCE Loss) for class prediction, combined with DFL loss [22] and CIOU loss [23] for bounding box coordinate regression.

#### 2.4.2. Anchor-Free

It directly predicts the target center point (x, y) and width and height (w, h), eliminating the need to adjust hyperparameters of preset anchor boxes [24]. The formula is:

$$\begin{aligned} x &= 2\sigma(t_x) - 0.5 + c_x, \quad y = 2\sigma(t_y) - 0.5 + c_y \\ w &= (2\sigma(t_w))^2 \times s_w, \quad h = (2\sigma(t_h))^2 \times s_h \end{aligned} \tag{1}$$

where , $c_x$,$c_y$ are grid coordinates and $s_w$,$s_h$ is the stride of the feature map.

## 3. Improved YOLOv8s Object Detection Algorithm

### 3.1. Cloformer and CBAM Attention Mechanism

Lightweight surface flaw identification models for hot-rolled steel strips often suffer from insufficient accuracy. At the same time, YOLOv8 also has certain issues in defect detection. Due to the limited sensitivity of its default architecture to defect features, it is prone to missed or false detections

 doi:10.20944/preprints202503.1303.v2



when facing lightweight and subtle surface defects. Moreover, this architecture is difficult to effectively integrate feature information of different scales, resulting in poor adaptability of the model in complex scenes. A common solution to this problem is to introduce an attention mechanism. CBAM addresses traditional CNN limitations in processing multi-scale, shape, and orientation information through dual attention mechanisms: channel attention improves feature representation across channels, while spatial attention extracts critical spatial information. CBAM comprises two key components: channel attention module (C-channel) and positional attention module (S-channel), which can be integrated at different network levels to strengthen feature representation. As shown in Figures 3 and 4, the operational principles of channel and spatial attention modules differ fundamentally. In CBAM, element-wise multiplication of both channel and positional attention mechanisms outputs produces enhanced attention features (Figure 5). These refined features propagate through subsequent layers, effectively preserving critical information while filtering noise.

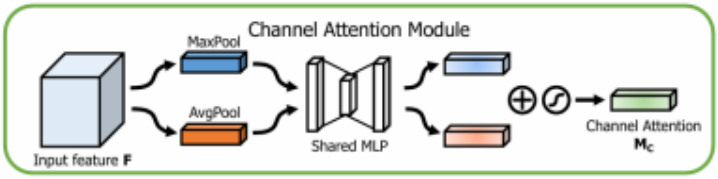


**Figure 3.** The principle of the channel attention module [12].

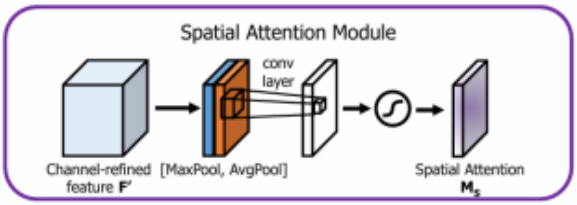


**Figure 4.** The principle of the spatial attention module [12].

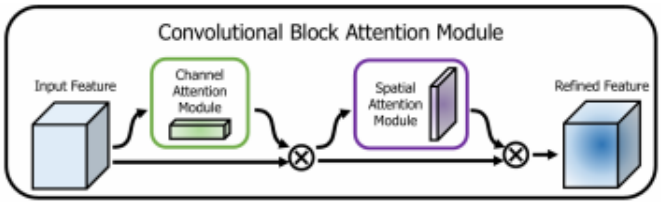


**Figure 5.** Hybrid Attention Module [12].

CloFormer introduces a module named AttnConv that combines the attention mechanism and convolution operation, capable of capturing high-frequency local information. Compared with traditional convolution operations, AttnConv employs shared weights and context-aware weighting to better process inter-position relationships. Experimental results demonstrate CloFormer's superior classification performance for images, object detection, and semantic segmentation challenges.

 doi:10.20944/preprints202503.1303.v2



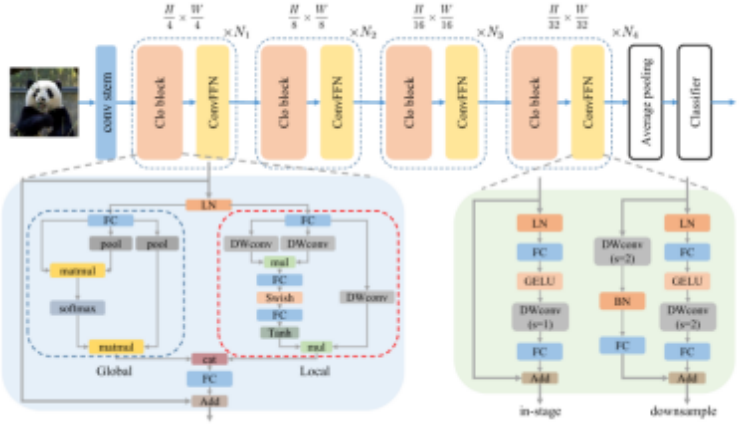


**Figure 6.** The illustration of CloFormer. CloFormer is constructed by connecting Clo block and ConvFFN in series. Alldepth-wise convolutions in the local branch have the stride of 1 [11].

Although CBAM is lightweight, its performance gains are modest. In contrast, while Cloformer can significantly boost accuracy, AttnConv employs shared weights and context-aware weighting to better process inter-position relationships. Experimental results demonstrate CloFormer's superior performance in classifying images, detecting objects, and segmenting semantically. However, this method substantially increases both model parameters and computational overhead. To optimize accuracy without substantially inflating the model size, this paper integrates both CBAM and Cloformer into the Neck module and investigates how various combinations affect precision and detection speed. As illustrated in Table 1, experiments 1-3 apply CBAM to the small, medium, and large object detection heads respectively, with the remaining heads receiving Cloformer input. Conversely, experiments 4-6 assign Cloformer to the small, medium, and large detection heads while using CBAM for the others. Scenarios that completely omit either CBAM or Cloformer are not considered, with the other two detection heads receiving input from CBAM. Scenarios that completely omit either CBAM or Cloformer were not considered in our experiments.

**Table 1.** The influence of CBAM and Cloformer at different positions on the model.

| Experiments | mAP@0.5 | Param(M) | FLOPs(G) | FPS |
|---|---|---|---|---|
| 1 | 76.8 | 12.69 | 15.17 | 98.8 |
| 2 | 77.4 | 12.51 | 15.48 | 94.6 |
| 3 | 77.5 | 11.81 | 15.52 | 89.1 |
| 4 | 77.1 | 11.57 | 15.09 | 102.3 |
| 5 | 76.6 | 11.75 | 14.79 | 102.9 |
| 6 | 76.1 | 12.45 | 14.74 | 102.7 |

Taking into account model size, detection speed, and accuracy, this paper adopts the combination strategy from Experiment 4. In this setup, the medium and large object detection heads utilize outputs from CBAM, while the small detection head uses Cloformer outputs. This configuration improves detection accuracy by 1.42% with an increase of only 0.4M parameters.

### *3.2. Model Lightweight Design*

Efforts to develop lightweight models for hot-rolled strip steel surface defect detection often involve replacing the backbone with networks such as MobileNet or ShuffleNet [17].The improved YOLOv8 architecture needs to be deployed on edge devices while ensuring detection accuracy. Edge devices have relatively limited computing power and memory resources, which requires the model to not only accurately identify lightweight surface defects in hot-rolled steel strips, but also have low computational complexity and memory usage. Therefore, it is necessary to incorporate lightweight design into the architecture design. However, this approach, while reducing computational load, it

 doi:10.20944/preprints202503.1303.v2



increases memory access overhead. Such approaches fail to effectively address network latency issues and may compromise inference speed, resulting in them being unsuitable for real-time surface defect detection in hot-rolled steel strips.

Depthwise separable convolution (DWConv) is a lightweight convolution operation method specifically designed to enhance memory access efficiency and operational speed.

Compared with standard convolution, depthwise separable convolution significantly reduces parameter count and computational load, alleviate overfitting, and improve inference speed and real-time detection performance. It has been proven to perform well in various lightweight models, substantially decreasing the computational and storage requirements of the model while maintaining consistent accuracy [25]. DWConv can significantly reduce the computational complexity by decomposing standard convolution into two blocks: depthwise convolution and pointwise convolution [26], as shown in Figures 7 and 8. For a convolution kernel of size $D_K \times \mathrm{D}_K$, input channels M, output channels N, and output feature map dimensions $D_F \times \mathrm{D}_F$, standard convolution parameters and computational costs are:

$$Params = D_K \times D_K \times M \times N \tag{2}$$

$$FLOPs = D_K \times D_K \times M \times N \times D_F \times D_F \tag{3}$$

The difference between depthwise convolution (DConv) and standard filtering process lies in that the depthwise convolution kernel is configured for single-channel operation, and convolution needs to be performed on each channel of the input. Thus, the output feature map will maintain the same channel dimension as the input feature map. That is, the number of input feature map channels = the number of output feature map channels. convolution kernels = the number of output feature map channels. Therefore, its parameter quantity and computational cost are:

$$Params = D_K \times D_K \times M \tag{4}$$

$$FLOPs = D_K \times D_K \times M \times D_F \times D_F \tag{5}$$

In depthwise convolution, identical channel counts for input features and convolution kernels may limit output feature diversity. Pointwise convolution (PWConv) addresses this through 1×1 convolution-based channel expansion, with parameters and computations being:

$$Params = M \times N \tag{6}$$

$$FLOPs = M \times N \times D_F \times D_F \tag{7}$$

Combining depthwise and pointwise convolutions reduces parameters and computations to approximately $\frac{1}{D_K^2}$ of standard convolution, particularly when N is large.

$$Params\ Ratio : \frac{D_K \times D_K \times M + M \times N}{D_K \times D_K \times M \times N} = \frac{1}{N} + \frac{1}{D_K^2} \tag{8}$$

$$FLOPs\ Ratio : \frac{D_K \times D_K \times M \times D_F \times D_F + M \times N \times D_F \times D_F}{D_K \times D_K \times M \times N \times D_F \times D_F} = \frac{1}{N} + \frac{1}{D_K^2} \tag{9}$$

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 April 2025 doi:10.20944/preprints202503.1303.v2



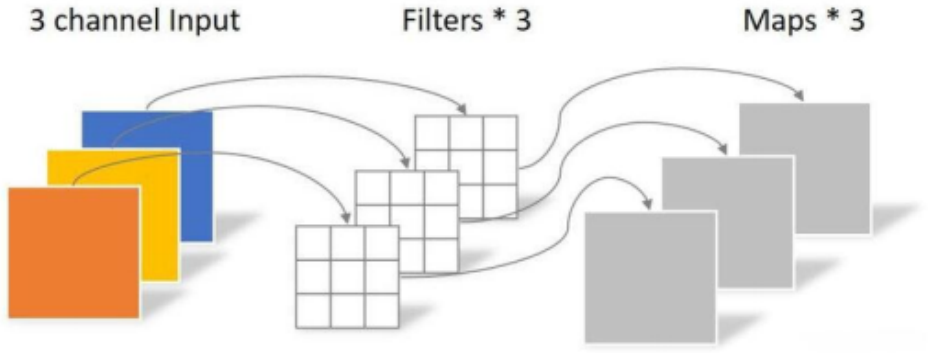


**Figure 7.** Depthwise Convolution.

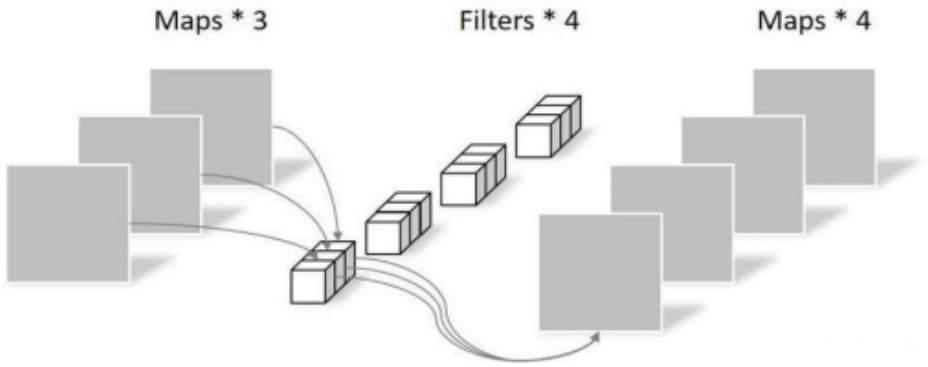


**Figure 8.** Pointwise Convolution.

In addition, we have observed a significant redundancy among different channels in the feature maps. Partial convolution (PConv) leverages channel redundancy by applying standard convolution to a continuous subset (e.g., first/last c channels) while keeping others unchanged. This maintains input-output dimensional consistency while reducing computation and parameters.

To ensure full channel information utilization, pointwise convolution (PWConv) immediately follows PConv. Together, these operations create a receptive field with significant impact that resembles a T-shaped convolution, which emphasizes the central region more than standard convolution. This decomposition into PConv and PWConv further exploits the redundancy between filters and reduces FLOPS, as illustrated in Figure 9.

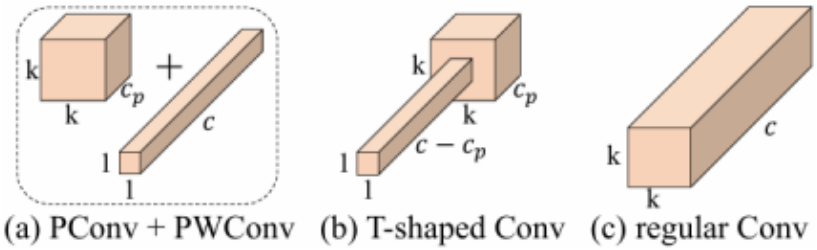


**Figure 9.** Comparison of convolutional variants. A PConv followed by a PWConv (a) resembles a T-shaped Conv (b), which spends more computation on the center position compared to a regular Conv (c) [14]

Experiments have demonstrated that PConv demonstrates effectiveness in spatial feature extraction. By constructing a simple network composed of PConv and PWConv and training it on a dataset of feature maps extracted from a pre-trained ResNet50, the results show that PConv + PWConv achieves the lowest test loss and better approximates the feature transformation of conventional convolution, indicating that capturing spatial features from only partial feature maps is sufficient and efficient [14].

 doi:10.20944/preprints202503.1303.v2



To integrate DWConv and PConv, this paper substitutes the standard convolution i in the backbone network by using DWConv and enhances the C2f module. Specifically, the conventional convolution operations in both the C2f module and the Bottleneck are substituted with a combination of PConv and PWConv, resulting in a notable drop in model parameters and improved accuracy. The architectures of the modified C2f module and Bottleneck are depicted in Figures 10 and 11.

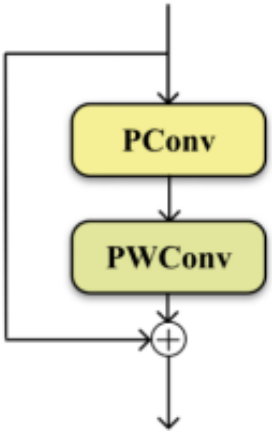


**Figure 10.** Bottleneck_PConv.

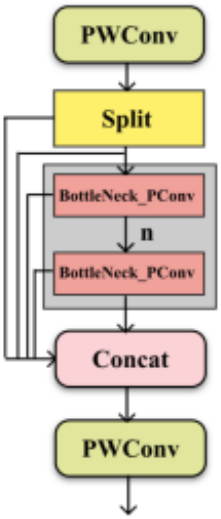


**Figure 11.** C2f_PConv

*3.3. Improvement of the SPPF Module*

Spatial Pyramid Pooling (SPP) is mainly designed to address the issue of variable input image sizes. In traditional convolutional neural networks, it is usually required that input images have a fixed size, which brings many inconveniences in practical applications. Spatial pyramid pooling generates fixed-length outputs for variable-size inputs, enabling flexible image processing. SPPF represents an optimized version of this approach [27]. However, while SPPF reduces the computational load, it increases the number of model parameters, which can have a certain impact on the training as well as the model's inference performance. The SimSPPF module is introduced to further accelerate model detection.

Originally proposed in YOLOv6, SimSPPF serves as a simplified spatial pyramid pooling structure for efficient feature extraction in computer vision tasks. As illustrated in Figure 12, SimSPPF comprises two primary components. The first component consists of a series of convolution operations, starting with an initial SimConv layer that processes the input feature map and reduces its channel count by half. SimConv is a custom module that integrates convolution, batch normalization, and ReLU activation; it generates features based on the input, accelerates training and stabilizes the model via

 doi:10.20944/preprints202503.1303.v2



batch normalization, and enhances expressive power through the nonlinearity provided by ReLU. The second component involves multiple max pooling and concatenation operations to fuse features from different scales, subsequently connected to a final SimConv layer that converts the fused feature map to the desired output channel number.

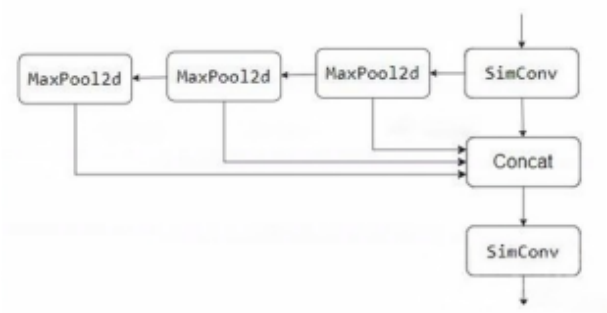


**Figure 12.** The structure of SimSPPF.

In order to enhance the inference speed of YOLOv8 model in steel defect detection, we replaced the SPPF module of the original model with SimSPPF module. The SimSPPF module has optimized the spatial pyramid pooling structure in its design. Compared to the SPPF module, it significantly reduces computational complexity by simplifying the operation process. In the process of steel defect detection, the complex surface texture and defect features require high computational efficiency of the model. The low computational load of SimSPPF module effectively reduces the computational burden of the model and accelerates the inference speed. After applying it to the surface defect detection of hot-rolled strip steel, the inference speed was increased by about 20%. In addition, the SimSPPF module exhibits good lightweight characteristics while maintaining model detection accuracy, which meets the deployment requirements on edge devices. Figure 13 shows the overall network architecture of the enhanced YOLOv8.

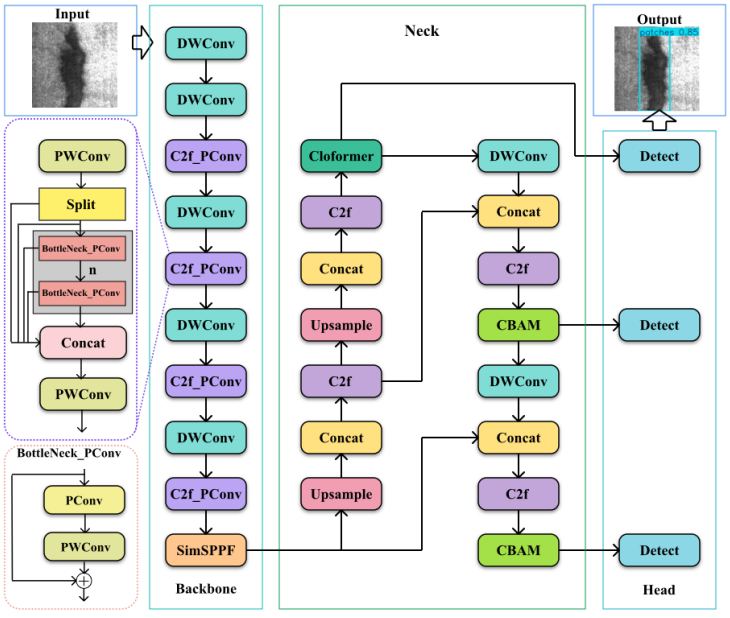


**Figure 13.** The optimized YOLOv8s network structure.

## 4. Experimental Results and Analysis

### *4.1. Experimental Dataset*

The NEU-DET public dataset for hot-rolled steel strips contains six defect types: Crazing (Cr), Inclusion (In), Patches (Pa), Pitted Surface (Ps), Rolled-in Scale (Rs), and Scratches (Sc). Each type

 doi:10.20944/preprints202503.1303.v2



includes 300 200×200 grayscale images, some containing multiple defects. Data is split 9:1 into training and test sets, with sample images shown in Figure 1.

### 4.2. Experimental Environment and Parameter Settings

Experiments employ an Intel Core i7-13700KF CPU with NVIDIA GeForce RTX 4060Ti GPU, running Windows 11, Python 3.10, PyTorch 2.6.0, and CUDA 12.6. In this paper, the initial learning rate for all YOLO models is 0.0003, which linearly decreases with the epochs, and the final learning rate is 1% of the initial value. The optimizer used is AdamW (Adam with Decoupled Weight Decay) [28]. A batch size of 8 is used, the number of training rounds is 500, and an early stopping mechanism is adopted with a patience of 80. All input images are uniformly adjusted to 640px × 640px, and data augmentation is performed using Albumentations [29].

### 4.3. Evaluation Indicators

Given multiple existing algorithms for hot-rolled steel strip defect detection, comprehensive evaluation metrics are essential for objective performance assessment.

This work employs conventional object detection metrics—such as model parameters (Params), computational complexity (FLOPs), defect-specific accuracy, we evaluate using Average Precision (AP), mean Average Precision (mAP), and processing speed (FPS)—to comprehensively assess performance. The Params and FLOPs metrics are used to evaluate the network's lightweight characteristics, while FPS measures its real-time detection capability. Generally, achieving an FPS above 30 indicates that the model meets the on-the-fly detection requirements.

Accuracy metrics include precision (P) and recall (R). Precision measures correct predictions among all detections, while recall indicates detected true defects. Calculation formulas are:

$$P = \frac{T_{\mathrm{P}}}{T_{\mathrm{P}} + F_{\mathrm{P}}} \quad R = \frac{T_{\mathrm{P}}}{T_{\mathrm{P}} + F_{\mathrm{N}}} \tag{10}$$

In Equation (10): $T_P$ represents the number of correctly predicted positive samples; $T_N$ represents the number of correctly predicted negative samples; $F_P$ represents the number of incorrectly predicted negative samples; $F_N$ represents the number of incorrectly predicted positive samples.

The calculation formulas for precision and average precision are:

$$AP = \int_0^1 p(R)\mathrm{d}R \quad mAP = \frac{\sum_{i=0}^{n} AP(i)}{n} \tag{11}$$

The performance of mAP varies at different thresholds. The most commonly used one is mAP@0.5, which represents the average precision of all categories when the threshold IOU is 0.5.

### 4.4. Melting Experiments and Result Analysis

To evaluate how these five enhancements influence the model's detection performance and complexity, we carried out ten experiments under consistent environmental and parameter conditions. Each experiment was trained five times, and the trial achieving the highest accuracy was recorded. Table 2 presents the outcomes, where "✓" denotes the incorporation of the corresponding module.

 doi:10.20944/preprints202503.1303.v2



**Table 2.** Ablation experiments on improvement points.

| Exp | CBAM | DWConv | PConv | CloFomer | SimSPPF | Params | FLOPs | P | R | mAP@0.5 | FPS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | / | / | / | / | / | 11.17 | 14.36 | 74.4 | 95 | 75.68 | 107.8 |
| 2 | ✓ | / | / | / | / | 11.51 | 14.36 | 74.3 | 95 | 76.65 | 102.4 |
| 3 | / | ✓ | / | / | / | 9.61 | 12.5 | 79.4 | 95 | 77.61 | 136.7 |
| 4 | / | / | ✓ | / | / | 8.31 | 10.35 | 71.1 | 97 | 75.23 | 173.5 |
| 5 | / | / | / | ✓ | / | 12.75 | 15.9 | 77.2 | 96 | 76.82 | 83.6 |
| 6 | / | / | / | / | ✓ | 11.17 | 14.36 | 76.8 | 94 | 75.88 | 129.9 |
| 7 | ✓ | ✓ | / | / | / | 9.95 | 12.5 | 74.9 | 96 | 78.43 | 122.6 |
| 8 | ✓ | / | ✓ | / | / | 8.62 | 10.35 | 73.5 | 96.5 | 76.9 | 168.3 |
| 9 | ✓ | / | / | ✓ | / | 12.98 | 15.9 | 76.8 | 95.8 | 77.2 | 85.2 |
| 10 | ✓ | / | / | / | ✓ | 11.51 | 14.36 | 75.2 | 95.5 | 76.4 | 115.6 |
| 11 | / | ✓ | ✓ | / | / | 7.85 | 10.35 | 75 | 96.8 | 76.1 | 155.4 |
| 12 | / | ✓ | / | ✓ | / | 10.2 | 13.2 | 78.6 | 96.2 | 78.9 | 105.7 |
| 13 | / | ✓ | / | / | ✓ | 9.61 | 12.5 | 78 | 95.5 | 77.8 | 130.5 |
| 14 | / | / | ✓ | ✓ | / | 10.03 | 12.8 | 75.3 | 97.2 | 76.5 | 142.1 |
| 15 | / | / | ✓ | / | ✓ | 8.31 | 10.35 | 72.8 | 96.8 | 75.9 | 165.2 |
| 16 | / | / | / | ✓ | ✓ | 12.75 | 15.9 | 77.8 | 96.5 | 77.6 | 90.4 |
| 17 | ✓ | ✓ | ✓ | / | / | 7.1 | 8.49 | 72.9 | 97 | 77.27 | 191.7 |
| 18 | ✓ | ✓ | / | ✓ | / | 10.75 | 13.2 | 77.5 | 97 | 79.2 | 110.3 |
| 19 | ✓ | ✓ | / | / | ✓ | 9.95 | 12.5 | 76.2 | 96.2 | 78.8 | 125 |
| 20 | ✓ | / | ✓ | ✓ | / | 9.35 | 12.8 | 76.8 | 97.5 | 78.5 | 135.8 |
| 21 | ✓ | / | ✓ | / | ✓ | 8.62 | 10.35 | 74 | 96 | 77.1 | 160.1 |
| 22 | / | ✓ | ✓ | ✓ | / | 9.1 | 11.2 | 78.2 | 97.5 | 78.7 | 148.9 |
| 23 | / | ✓ | ✓ | / | ✓ | 7.85 | 10.35 | 75.6 | 96.3 | 76.8 | 157.3 |
| 24 | ✓ | ✓ | ✓ | ✓ | / | 8.27 | 9.31 | 77 | 98 | 79.08 | 180.7 |
| 25 | ✓ | ✓ | ✓ | ✓ | ✓ | 8.28 | 9.31 | 77.5 | 98 | 80.1 | 182.4 |

Table 2 indicates that augmenting YOLOv8s with either CBAM or Cloformer substantially raises the average detection accuracy for identifying defects on hot-rolled steel strips. Conversely, introducing DWConv lowers the model's parameter count and computational overhead, thereby markedly increasing detection speed while also improving accuracy. Although using partial convolution results in a slight 0.45% decrease in accuracy, its major benefits in lowering parameters and computational cost—and the corresponding speed gains—are considerable. Additionally, the combination with SimSPPF not only improves detection accuracy marginally but also increases inference speed by approximately 20%. Ultimately, the fully enhanced YOLOv8s model (YOLOv8s+CBAM+DWConv+PConv+Cloformer+SimSPPF) achieves superior performance in detection accuracy, precision, and recall. Its outstanding detection speed is particularly critical for industrial defect detection scenarios with high real-time demands. Therefore, the proposed improved algorithm delivers both high accuracy and exceptional real-time performance in detecting irregularities on the surface of hot-rolled steel strips, proving its effectiveness in practical applications.

### *4.5. Controlled Experiment*

#### 4.5.1. Comparison Before and After Improvement

To assess the effectiveness of the algorithm in detection tasks RM-YOLOv8 proposed in this paper on various types of surface imperfections in steel strips, the average precision of the original YOLOv8 and RM-YOLOv8 in detecting various types of defects on hot-rolled steel strips was compared. The results of the comparison are illustrated in Figure 14.

As depicted in Figure 14, the proposed algorithm demonstrates effective detection across all defect types.

Among them, the mAP of crazing defects increased by 8.6%, the mAP of rolled-in scale defects rose by 4.62%, and the mAP of the remaining defect types also increased by approximately 3–4% each.

Furthermore, to assess the attention module's focus on the target, To validate defect localization capability, Grad-CAM++ [30] visualization is employed. Figure 15 shows brighter, concentrated color regions indicating accurate defect positioning.

Based on the heat map in Figure 15, incorporating both the CBAM and Cloformer attention mechanisms noticeably enhances the model's focus on steel strip defects, the improved YOLOv8s

 doi:10.20944/preprints202503.1303.v2



model exhibits a stronger response at the steel strip's defect locations compared to the original YOLOv8s.

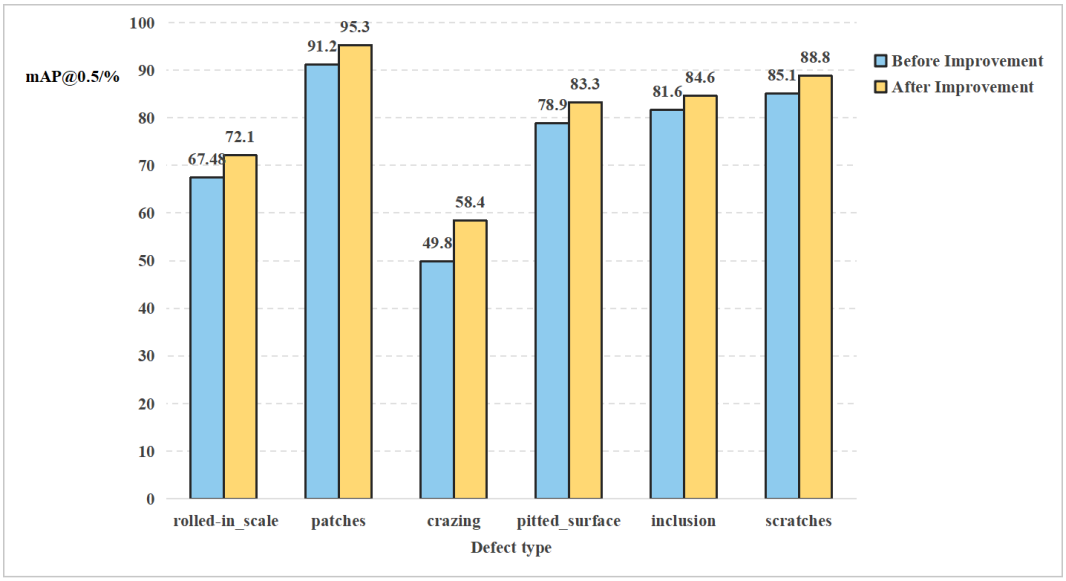


**Figure 14.** Comparison of average detection accuracy of various defects before and after improvement.

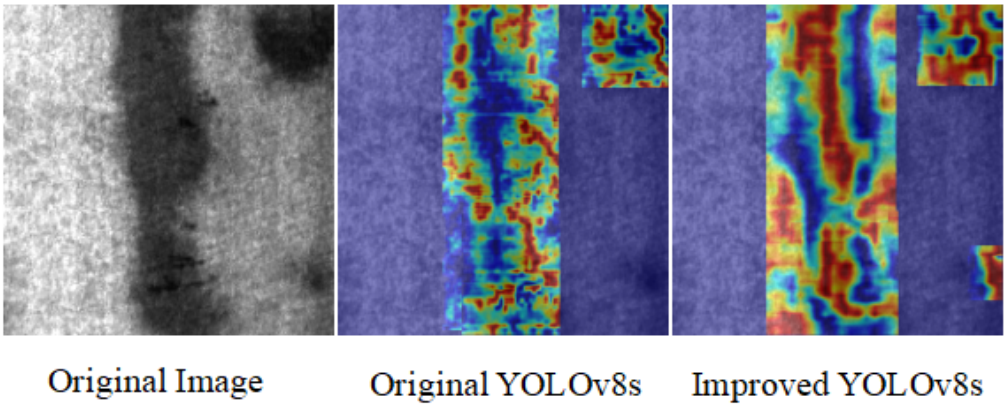


**Figure 15.** Comparison of heatmap results before and after the improvement of the YOLOv8 model.

4.5.2. Comparison with Other Algorithms

Further validation of the improved algorithm's detection performance is conducted, a comprehensive comparison was conducted on the NEU-DET dataset under consistent experimental conditions and parameter settings. The evaluation included representative models such as SSD [31], Faster R-CNN, EfficientDet-D1 [32], RT-DETR-1 [33], GoldYOLO-n [34],along with classic and latest YOLO series models. The experimental results are detailed in Table 3.

Comprehensive evaluations highlight that the enhanced YOLOv8s achieves optimal compromise between detection accuracy and computational efficiency on the NEU-DET dataset. When benchmarked against the conventional two-stage Faster R-CNN, our enhanced YOLOv8s model achieves a 4.01% increase in mAP, cuts down parameters by 80.2%, and delivers an 8.3-fold improvement in inference speed. Moreover, compared to YOLOv5s, it attains a 4.8% higher detection accuracy while keeping the inference speed nearly unchanged.Although GoldYOLO-n excels in lightweight design, its mAP50 is still lower than that of our model.Notably, RT-DETR-1 based on Transformer, despite having a comparable accuracy about 80.3%, has significantly higher computational cost and slower inference speed than our model, making it difficult to meet the real-time detection requirements in industrial applications. Additionally, the community-improved models YOLOv12-n and YOLOv11-n, despite having extremely low parameter counts, neither surpass our model in terms of accuracy nor speed, thus verifying the necessity of the modular optimization in this study. These results confirm

 doi:10.20944/preprints202503.1303.v2



the proposed method's advantages in industrial defect detection: high precision with low resource consumption.

**Table 3.** Performance Comparison of Various Algorithms.

| Method | mAP@0.5 | FLOPs | Params | FPS |
|---|---|---|---|---|
| YOLOv5s | 75.6 | 7.2 | 8.7 | 195 |
| YOLOv5n | 71.2 | 4.5 | 1.9 | 280 |
| YOLOv6s | 78.2 | 9.8 | 12.4 | 165 |
| YOLOv6n | 73.5 | 6.1 | 4.2 | 220 |
| YOLOv7 | 79.1 | 13.5 | 37.4 | 135 |
| YOLOv8n | 74.8 | 5.8 | 3.1 | 250 |
| YOLOv8s | 77.3 | 8.4 | 11.2 | 190 |
| YOLOv8m | 79.5 | 15.2 | 25.8 | 120 |
| YOLOv8l | 80.7 | 21.9 | 43.7 | 85 |
| YOLOv9-t | 77.8 | 7.6 | 9.8 | 175 |
| YOLOv11-n | 74.5 | 6.6 | 2.6 | 240 |
| YOLOv12-n | 76.2 | 6.7 | 2.6 | 235 |
| YOLOv10-n | 76.4 | 5.2 | 2.5 | 265 |
| GoldYOLO-n | 78.5 | 12.6 | 5.6 | 200 |
| RT-DETR-1 | 80.3 | 103 | 32 | 77 |
| Faster R-CNN | 76.4 | 41.5 | 134.2 | 22 |
| SSD | 68.9 | 26.8 | 35.4 | 125 |
| EfficientDet-D1 | 73.8 | 13.6 | 10.5 | 95 |
| **Ours** | **80.41** | **8.21** | **9.31** | **182.4** |

## 5. Discussion

The excellent performance of the improved YOLOv8s on the NEU-DET dataset validates the effectiveness of lightweight design and task-specific optimization. As opposed to YOLOv5s, although model parameter size has slightly increased, the mAP has improved by 4.8%, mainly owing to the implementation of the Cloformer and CBAM modules, which significantly enhance the model's ability to locate tiny cracks. Meanwhile, by introducing DWConv and PConv, the model significantly reduces redundant computations while retaining the ability to extract key features. The number of parameters is 80.2% less than that of Faster R-CNN, and the computational cost is only 26.3% of SSD300. However, it was observed that when defect size falls below 10×10 pixels (as with crazing defects), detection accuracy drops by around 20%. This decline is likely due to the low-resolution limitations of the feature pyramid at the lower levels. Moreover, the deployment efficiency of the model on embedded devices has not been verified and still requires further experimental verification and optimization. Future research will explore the joint optimization of super-resolution reconstruction and detection tasks to further enhance the sensitivity to microscopic defects.

## 6. Conclusions

Addressing lightweight network design and small target recognition challenges in hot-rolled steel strip defect detection, this study presents an optimized YOLOv8s model. Results show significant improvements over the original model in accuracy, model size, and inference speed. Moreover, when measured against current mainstream surface defect detection algorithms, the improved YOLOv8s not only boosts detection speed but also offers a lightweight design that is well-suited for edge deployment in industrial scenarios. These results provide valuable insights and serve as a reference for advancing surface defect detection in hot-rolled strip steel.

**Author Contributions:** Conceptualization,B.T.Q.; methodology,B.T.Q.; software,B.T.Q.; validation, B.T.Q.; resources,B.T.Q.; writing—original draft preparation,B.T.Q.; writing—review and editing,B.T.Q.; visualization, B.T.Q.; supervision,B.T.Q.; project administration,B.T.Q.; All authors have read and agreed to the published version of the manuscript.

 doi:10.20944/preprints202503.1303.v2



**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Data are contained within the article. The data presented in this study can be requested from the authors.

**Acknowledgments:** When reviewing and preparing the methods mentioned in this manuscript, the deepseek artificial intelligence language model was utilized to enhance the readability and grammatical accuracy of the text. The authors reviewed and revised the generated content to ensure its accuracy and consistency with the original work.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Fang, F.; Hu, X.-j.; Zhang, B.-m.; Xie, Z.-h.; Jiang, J.-q. Deformation of dual-structure medium carbon steel in cold drawing. *Materials Science and Engineering: A* **2013**, *583*, 78–83. doi:https://doi.org/10.1016/j.msea.2013.06.081
2. Zhou, L.; Gong, J.; Li, B. Image Information Restoration of Automotive Strip Steel Surface Based on Sparse Representation. *Hunan Daxue Xuebao/Journal of Hunan University Natural Sciences* **2021**, *48*, 141–148. doi:10.16339/j.cnki.hdxbzkb.2021.08.018
3. Redmon, J.; Divvala, S.K.; Girshick, R.B.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* **2015**, 779–788.
4. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2017**, *39*, 1137–1149. doi:10.1109/TPAMI.2016.2577031
5. Feng, X.; Gao, X.-w.; Luo, L. X-SDD: A New Benchmark for Hot Rolled Steel Strip Surface Defects Detection. *Symmetry* **2021**, *13*, 706.
6. Li, S.; Wang, X. YOLOv5-based Defect Detection Model for Hot Rolled Strip Steel. *Journal of Physics: Conference Series* **2022**, *2171*, 012040. doi:10.1088/1742-6596/2171/1/012040
7. Shen, L.; Shi, T.; Liao, J. Surface Defect Detection Algorithm of Hot-Rolled Strip Based on Improved YOLOv7. IAENG International Journal of Computer Science 2024, 51, 345-354.
8. Zhang, W.K.; Liu, J. Steel Surface Defect Detection Based on Improved YOLOv8s. *Journal of Beijing Information Science & Technology University (Natural Science Edition)* **2023**, 33–40. doi:doi:10.16508/j.cnki.11-5866/n.2023.06.005
9. Wang, L.L.; Gong, Z.Z.; Liang, Z.Q. Surface Defect Detection of Strip Steel Based on Improved YOLOv5s Algorithm. *Machine Tool & Hydraulics* **2024**, 181–186. doi:doi:10.13462/j.cnki.mmtamt.2024.12.034
10. Dong, J.; Cheng, J.; Wu, J.; Zhang, C.; Zhao, S.; Tang, X. Real-Time AIoT for UAV Antenna Interference Detection via Edge-Cloud Collaboration. *IEEE Internet of Things Journal* **2024**, 1-1. doi:10.1109/JIOT.2024.3512867
11. Fan, Q.; Huang, H.; Guan, J.; He, R. Rethinking Local Perception in Lightweight Vision Transformer. **2023**.
12. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. CBAM: Convolutional Block Attention Module. *Springer, Cham* **2018**.
13. Chollet, F. Xception: Deep Learning with Depthwise Separable Convolutions. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 21-26 July 2017, 2017; pp. 1800-1807.
14. Chen, J.; Kao, S.h.; He, H.; Zhuo, W.; Wen, S.; Lee, C.H.; Chan, S.H.G. Run, Don't Walk: Chasing Higher FLOPS for Faster Neural Networks. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 17-24 June 2023, 2023; pp. 12021-12031.
15. Li, C.; Li, L.; Jiang, H.; Weng, K.; Geng, Y.; Li, L.; Ke, Z.; Li, Q.; Cheng, M.; Nie, W.; et al. YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications. *ArXiv* **2022**, *abs/2209.02976*.
16. Hussain, M. YOLO-v1 to YOLO-v8, the Rise of YOLO and Its Complementary Nature toward Digital Manufacturing and Industrial Defect Detection. *Machines* **2023**.
17. Zhu, X.H.; Li, G.W.; Chang, D.F.; Du, J.W. A Surface Defect Detection Method for Hot-Rolled Strip Steel Based on Improved YOLOv8s. *Ship Engineering* **2025**, 124–131. doi:doi:10.13788/j.cnki.cbgc.2025.01.16
18. Bochkovskiy, A.; Wang, C.-Y.; Liao, H.-Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *ArXiv* **2020**, *abs/2004.10934*.

 doi:10.20944/preprints202503.1303.v2




19. Sun, Y.; Chen, G.; Zhou, T.; Zhang, Y.; Liu, N. Context-aware Cross-level Fusion Network for Camouflaged Object Detection. In Proceedings of the International Joint Conference on Artificial Intelligence, 2021.
20. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 21-26 July 2017, 2017; pp. 936-944.
21. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path Aggregation Network for Instance Segmentation. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 18-23 June 2018, 2018; pp. 8759-8768.
22. Li, X.; Wang, W.; Wu, L.; Chen, S.; Hu, X.; Li, J.; Tang, J.; Yang, J. Generalized Focal Loss: Learning Qualified and Distributed Bounding Boxes for Dense Object Detection. *ArXiv* **2020**, *abs/2006.04388*.
23. Zheng, Z.; Wang, P.; Liu, W.; Li, J.; Ye, R.; Ren, D. Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression. *ArXiv* **2019**, *abs/1911.08287*.
24. Huang, L.; Yang, Y.; Deng, Y.; Yu, Y. DenseBox: Unifying Landmark Localization with End to End Object Detection. *ArXiv* **2015**, *abs/1509.04874*.
25. Wu, C.K.; Huang, F.; Li, B.; Gu, L.L.; Liu, L.; Fang, Y.M. DMS-YOLOv8 Slab Detection Algorithm Based on Improved Depthwise Separable Convolution and Hybrid Attention Mechanism. *Metallurgical Automation* **2024**, 31–39.
26. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *ArXiv* **2017**, *abs/1704.04861*.
27. Jocher, G.; Stoken, A.; Chaurasia, A.; Borovec, J.; Kwon, Y.; Michael, K.; ... Thanh Minh, M. ultralytics/yolov5: v6.0-YOLOv5n 'Nano' models, Roboflow integration, TensorFlow export, OpenCV DNN support. *Zenodo* **2021**.
28. Loshchilov, I.; Hutter, F. Decoupled Weight Decay Regularization. In Proceedings of the International Conference on Learning Representations, 2017.
29. Buslaev, A.V.; Parinov, A.; Khvedchenya, E.; Iglovikov, V.I.; Kalinin, A.A. Albumentations: fast and flexible image augmentations. *ArXiv* **2018**, *abs/1809.06839*.
30. Chattopadhay, A.; Sarkar, A.; Howlader, P.; Balasubramanian, V.N. Grad-CAM++: Generalized Gradient-Based Visual Explanations for Deep Convolutional Networks. In Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), 12-15 March 2018, 2018; pp. 839-847.
31. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.E.; Fu, C.-Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the European Conference on Computer Vision, 2015.
32. Tan, M.; Pang, R.; Le, Q.V. EfficientDet: Scalable and Efficient Object Detection. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 13-19 June 2020, 2020; pp. 10778-10787.
33. Zhao, Y.; Lv, W.; Xu, S.; Wei, J.; Wang, G.; Dang, Q.; Liu, Y.; Chen, J. DETRs Beat YOLOs on Real-time Object Detection. In Proceedings of the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2024, pp. 16965–16974.
34. Wang, C.; He, W.; Nie, Y.; Guo, J.; Liu, C.; Wang, Y.; Han, K. Gold-YOLO: Efficient Object Detector via Gather-and-Distribute Mechanism. In Proceedings of the Advances in Neural Information Processing Systems; Oh, A.; Naumann, T.; Globerson, A.; Saenko, K.; Hardt, M.; Levine, S., Eds. Curran Associates, Inc., 2023, Vol. 36, pp. 51094–51112