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

Bingtian Qiao *

Posted Date: 18 March 2025

doi: 10.20944/preprints202503.1303.v1

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

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 real-time industrial applications. This paper proposes an optimized YOLOv8s algorithm for defect detection on hot-rolled steel strips. By integrating Cloformer and CBAM attention mechanisms, the model enhances its ability to capture spatial relationships, while the combination of partial convolution and depthwise separable convolution substantially reduces computational load. Furthermore, the introduction of SimSPPF accelerates inference. Experimental results show that the optimized YOLOv8s achieves a mean average precision (mAP) of 80.4% and an inference speed of 182.4 FPS, with a significant reduction in model size compared to conventional methods.

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, the development of precise and efficient defect detection methods is essential for manufacturers to ensure rigorous quality control and enhance overall safety.

With the rise of the Internet and the vigorous development of deep learning technology, more and more machine vision tasks adopt deep learning models. Compared with traditional machine vision technology, machine vision defect detection based on deep learning has higher accuracy, greater adaptability and a wider range of applications.

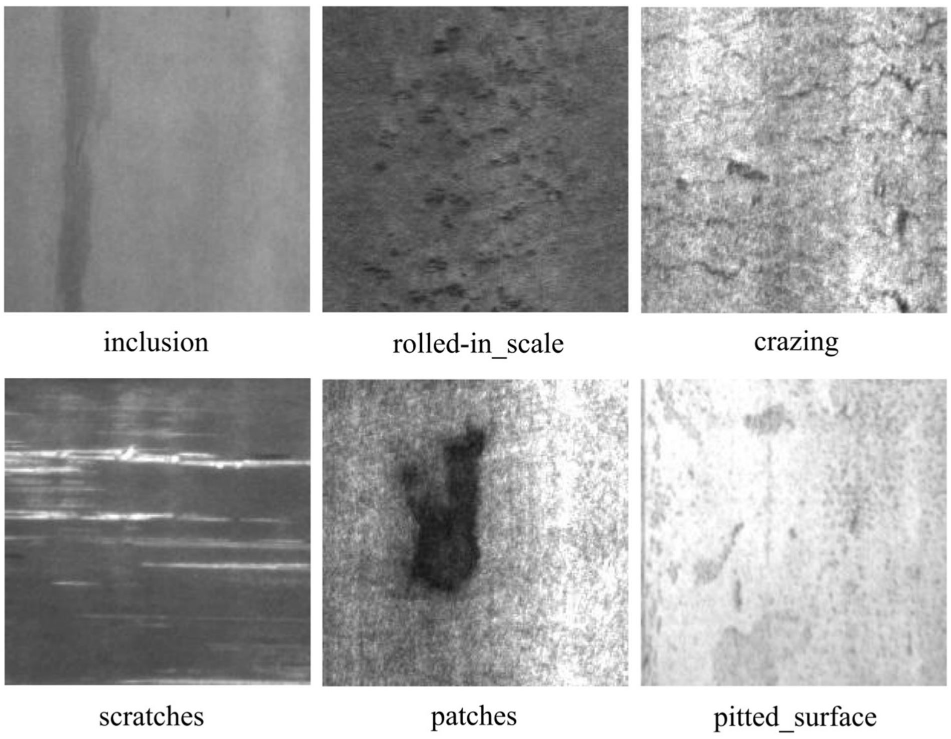

Figure 1. Six types of surface defects of hot-rolled strip steel.

A common deep learning object detection method is the YOLO series technology. YOLOv1 [3] was proposed by Joseph in 2015 and is a pioneering single-stage deep learning detection algorithm that improves the speed of object detection while maintaining a certain level of accuracy. As a representative single-stage method, YOLO directly generates the existence probability and position coordinates of the detected objects during the detection process. Its characteristic is a relatively fast speed but at the expense of detection accuracy. In contrast, the two-stage method represented 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 achieve a balance between detection accuracy and speed, Feng et al. [5] improved the RepVGG algorithm and combined it with the Spatial Attention (SA) mechanism. Shun et al. [6] proposed a YOLOv5-based method for detecting surface defects on hot-rolled steel strips and demonstrated that it significantly improves accuracy compared to YOLOv4. [7] introduced the BI-SPPFCSPC structure into the feature pyramid of YOLOv7 to enhance the feature extraction of small objects. 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, it dramatically decreased the number of model parameters while preserving a high level of 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.

In order to meet the stringent requirements for real-time detection and high accuracy in identifying surface defects on hot-rolled steel strips, this study improves the YOLOv8s network by focusing on three aspects: enhancing detection precision, reducing model complexity, and accelerating inference speed. 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, which not only improves the model's ability to recognize objects of varying sizes but also accelerates the 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 is available in five variants: YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x. For detecting surface defects on hot-rolled steel strips, this study selects the YOLOv8s model as the foundation to balance detection accuracy, speed, and a moderate model size. The YOLOv8s architecture is primarily divided into four components: the input module, Backbone, Neck, and Head, as illustrated in Figure 2.

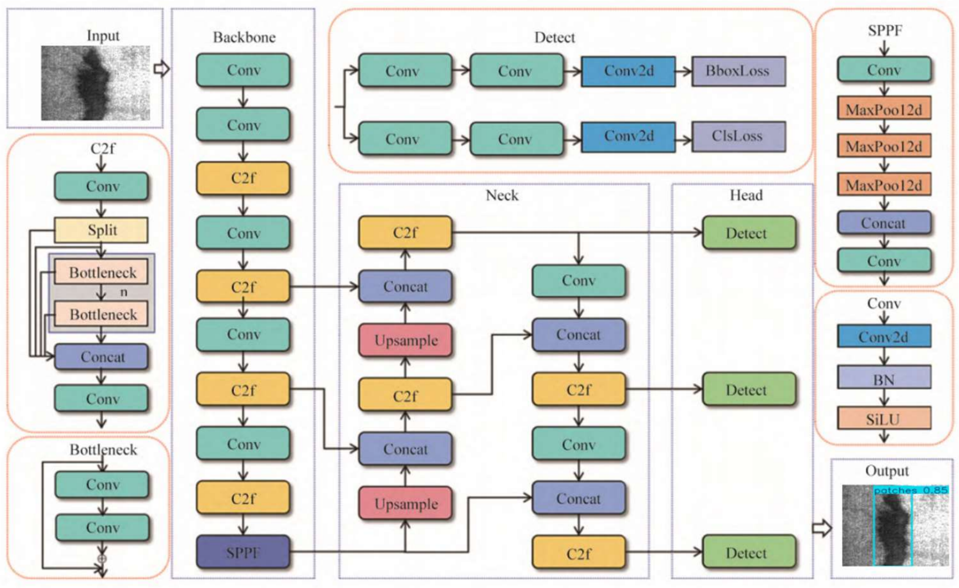

Figure 2. YOLOv8s Network Structure Diagram [17].

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.

2.2. Backbone

Backbone is responsible for feature extraction. Its core modules include the C2f module [19] (Cross Stage Partial with 2 convolutions and fusion), the SPPF module (Spatial Pyramid Pooling Fast), and the basic convolution module (Conv Block).

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. The process begins by compressing the input channels with a 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 parameters by about 20% while enhancing 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. This can expand the receptive field and enhance the adaptability to targets of different sizes.

2.3. Neck

The Neck module is tasked with fusing multi-scale features using an enhanced PAN-FPN architecture. In this design, the FPN pathway conveys high-level semantic information from smaller feature maps downwards, while the PAN 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 head uses BCE Loss to predict the category probability and a combination of DFL Loss [22] and CIOU Loss [23] to predict the offset of the center point and the width and height of the bounding box.

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:

$$x = 2\sigma(t_x) - 0.5 + c_x, \quad y = 2\sigma(t_y) - 0.5 + c_y$$

$$w = \left(2\sigma(t_w)\right)^2 \times s_w, \quad h = \left(2\sigma(t_h)\right)^2 \times s_h \tag{1}$$

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 defect detection models for hot-rolled strip steel often suffer from insufficient accuracy. A common solution to this problem is to introduce an attention mechanism. CBAM aims to overcome the limitations of traditional convolutional neural networks in handling information of different scales, shapes, and directions. To this end, CBAM introduces two attention mechanisms: channel attention and spatial attention. Channel attention helps enhance the feature representation of different channels, while spatial attention helps extract key information at different positions in the space. CBAM consists of two key components: the channel attention module (C-channel) and the spatial attention module (S-channel). These two modules can be integrated into various layers of the CNN to bolster feature representation. As depicted in Figures 3 and 4, the channel attention module and the spatial attention module operate on distinct principles. In CBAM, the outputs from these modules are multiplied element-wise to produce the final attention-enhanced features (see Figure 5). These enhanced features are then passed to subsequent network layers, effectively preserving essential information while filtering out noise and irrelevant details.

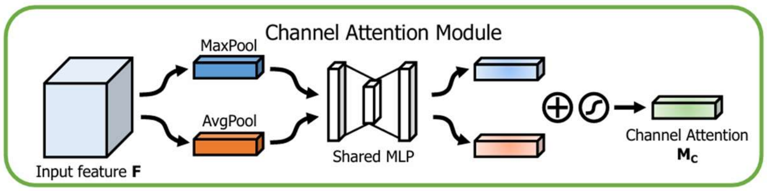

Figure 3. The principle of the channel attention module [12].

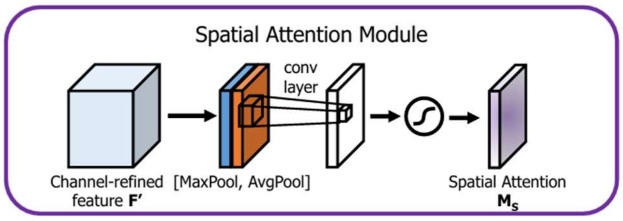

Figure 4. The principle of the spatial attention module [12].

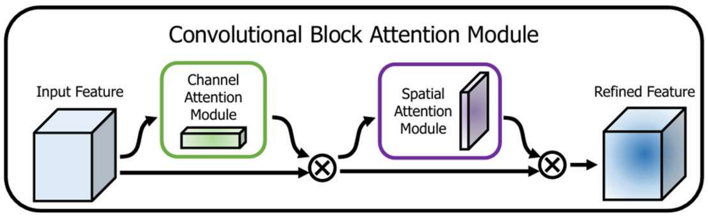

Figure 5. Hybrid Attention Module [12].

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 uses shared weights and context-aware weights, which can better handle the relationships between different positions in the image. Experimental results indicate that CloFormer consistently outperforms in image classification, object detection, and semantic segmentation tasks.

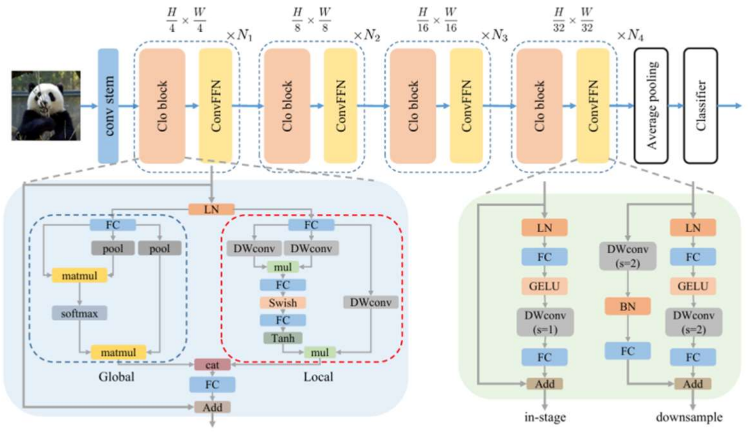

Figure 6. The illustration of CloFormer. CloFormer is constructed by connecting Clo block and ConvFFN in series. Alldepth-wise convolutions in the local branch have the stride of 1 [11].

Although CBAM is lightweight, its performance gains are modest. In contrast, while Cloformer can significantly boost accuracy, it also greatly increases the model's parameter count 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 accuracy and detection speed. As shown 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	Params(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 **Error! Reference source not found.**. However, this approach, while reducing computational load, it increases memory access overhead. This approach not only fails to solve network latency issues but also slows down the model's inference speed, hindering real-time detection of surface defects on hot-rolled strip steel.

Depthwise separable convolution (DWConv) is a lightweight convolution operation method specifically designed to enhance memory access efficiency and operational speed. Compared with ordinary convolution, depthwise separable convolution can effectively reduce the number of parameters 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, significantly reducing the computational and storage requirements of the model while maintaining high accuracy [25]. DWConv can significantly reduce the computational complexity by decomposing standard convolution into two steps: depthwise convolution and pointwise convolution [26], as shown in Figure 7 and Figure 8. Suppose the size of the convolution kernel is D_K*D_K, the number of input channels is M, the number of output channels is N, and the size of the output feature map is D_F*D_F, then the number of parameters and the computational cost of the ordinary convolution are:

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

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

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

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

$$\text{FLOPs} = D_K * D_K * M * D_F * D_F \tag{5}$$

In depthwise convolution, the number of input feature map channels is equal to both the number of convolution kernels and the number of output feature maps. This can result in an insufficient number of output feature maps, which may compromise the integrity of the information. To counteract this, a pointwise convolution is applied. Pointwise Convolution (PWConv) essentially serves as a dimensionality expansion operation using a 1×1 convolution kernel. Consequently, its parameter count and computational complexity are:

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

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

By combining depthwise convolution with pointwise convolution, depthwise separable convolution can reduce the parameter count and computational cost to roughly 1/(D_K²) of that of a standard convolution, particularly when N is large.

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

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

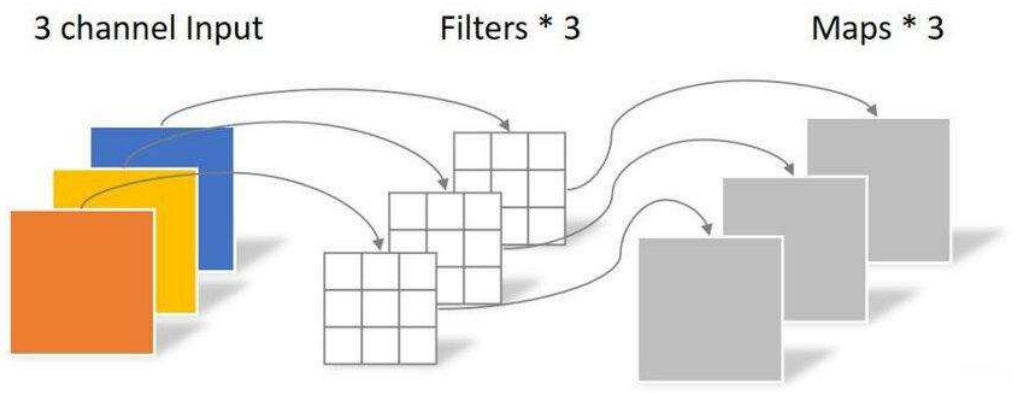

Figure 7. Depthwise Convolution.

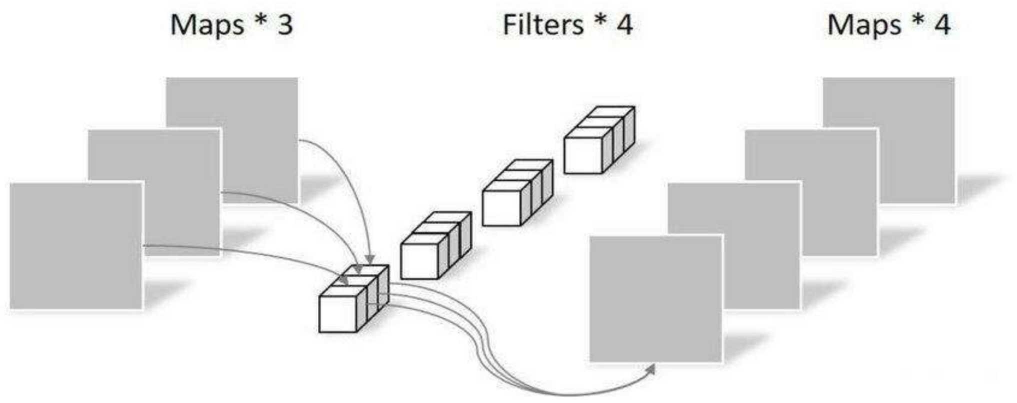

Figure 8. Pointwise Convolution.

In addition, we have observed a significant redundancy among different channels in the feature maps. Partial Convolution (PConv) leverages this redundancy by applying regular convolution to only a selected subset of consecutive channels—such as the first or last c channels—while leaving the remaining channels unchanged. This approach reduces both the computational cost and the number of parameters. The output feature map maintains the same dimensions as the input.

To ensure that information from all channels is fully utilized, a pointwise convolution (PWConv) is applied immediately after PConv. Together, these operations create an effective receptive field 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.

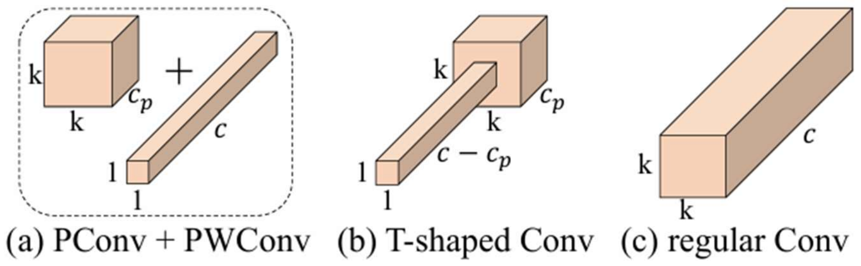

Figure 9. Comparison of convolutional variants. A PConv followed by a PWConv (a) resembles a T-shaped Conv (b), which spends more computation on the center position compared to a regular Conv (c) [14].

Experiments have demonstrated that PConv is effective in extracting spatial features. 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].

To integrate DWConv and PConv, this paper replaces the standard convolution in the backbone with 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 significant reduction in model parameters and improved accuracy. The architectures of the modified C2f module and Bottleneck are depicted in Figures 10 and 11.

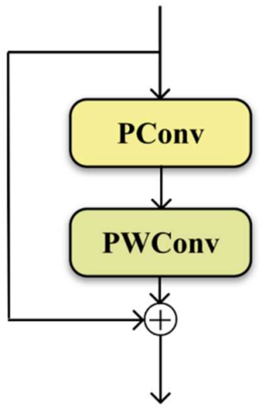

Figure 10. Bottleneck_PConv.

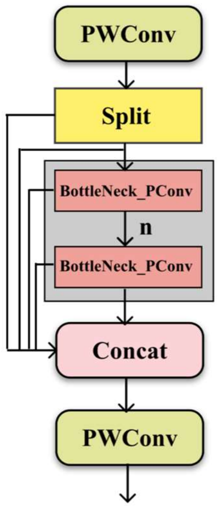

Figure 11. C2f_PConv.

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 can generate fixed-length outputs for input images of different sizes, enabling the network to accept images of any size. SPPF (Spatial Pyramid Pooling - Fast) is an improved version of spatial pyramid pooling [27]. However, while SPPF reduces the computational load, it increases the number of model parameters, which can have a certain impact on the training and inference speed of the model. To further enhance the detection speed of the model, this paper introduces SimSPPF.

The SimSPPF (Simplified Spatial Pyramid Pooling Fast) module, introduced in YOLOv6, is a streamlined spatial pyramid pooling approach designed for 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 extracts features from the input, accelerates training and stabilizes the model via 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, followed by a final SimConv layer that converts the fused feature map to the desired output channel number.

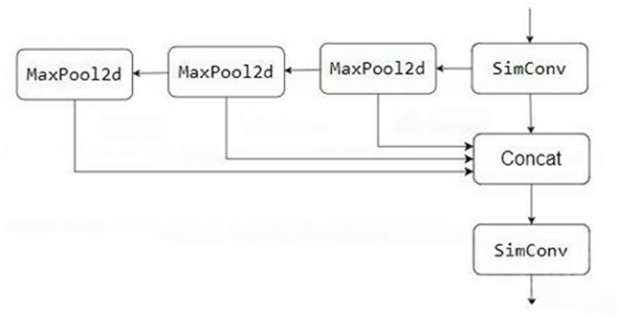

Figure 12. The structure of SimSPPF.

Replacing the original model's SPPF module with the SimSPPF module resulted in an approximate 20% boost in inference speed. Figure 13 presents the overall network architecture of the enhanced YOLOv8s.

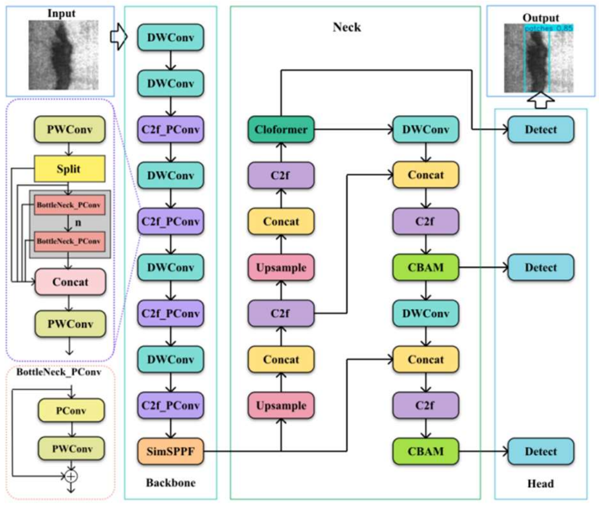

Figure 13. The optimized YOLOv8s network structure.

4. Experimental Results and Analysis

4.1. Experimental Dataset

The dataset used in this paper is the public dataset NEU-DET for hot-rolled steel strip. The dataset contains six types of defects, including crazing (Cr), inclusion (In), patches (Pa), pitted surface (Ps), rolled-in scale (Rs), and scratches (Sc). There are 300 grayscale images for each of the six types of steel surface defects, with a pixel size of 200px × 200px. Some images have more than one defect.

We randomly divided them into a training set and a test set at a ratio of 9:1. Examples of the dataset are shown in Figure 1.

4.2. Experimental Environment and Parameter Settings

The hardware environment of this experiment is an Intel Core i7-13700KF, with a GPU of NVIDIA GeForce RTX 4060Ti. The software environment includes Windows 11, Python (version 3.10), PyTorch (version 2.6.0), and CUDA (version 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]. The batch size is set to 8, the number of training rounds is 500, and an early stopping mechanism is adopted with a patience of 80. The input image size is uniformly scaled to 640px × 640px, and data augmentation is performed using Albumentations [29].

4.3. Evaluation Indicators

There are numerous algorithms for detecting surface defects on hot-rolled strip steel, making it essential to use a comprehensive set of evaluation metrics to assess performance. This work employs conventional object detection metrics—such as model parameters (Params), computational complexity (FLOPs), defect-specific accuracy, average precision (AP), mean average precision (mAP), and frames per second (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 real-time detection requirements.

Accuracy is used to evaluate the proportion of surface defects of hot-rolled strip steel correctly detected in all detection boxes. It is usually measured by precision (P) and recall (R). Precision, also known as the positive predictive value, is the proportion of correct samples in the predicted results; Recall, also known as the true positive rate, is the proportion of detected samples. The calculation formulas for precision P and recall R are:

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

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:

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

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.

Table 2. Ablation experiments on improvement points.

Experiment	CBAM	DWConv	PConv	CloFomer	SimSPPF	Params	FLOPs	P	R	mAP50	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	√	√	√	/	/	7.10	8.49	72.9	97	77.27	191.7
9	√	√	√	√	/	8.27	9.31	77.2	98	79.08	180.7
10	√	√	√	√	√	8.28	9.31	77.5	98	80.41	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 surface defects on hot-rolled steel strips, proving its effectiveness in practical applications.

4.5. Controlled Experiment

4.5.1. Comparison Before and After Improvement

To verify the detection effect of the algorithm RM-YOLOv8 proposed in this paper on various types of surface defects of 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 comparison results are shown in Figure 14.

As can be seen from Figure 14, the algorithm proposed in this paper has good detection accuracy for all types of defects. 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, Gradient-weighted Class Activation Map++ (Grad-CAM++) [30] was used to visualize the defect locations.. When the model predicts a defect, a brighter, more concentrated color at the defect location indicates a more accurate localization by the model. The comparison effect of the heat map is shown in Figure 15.

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

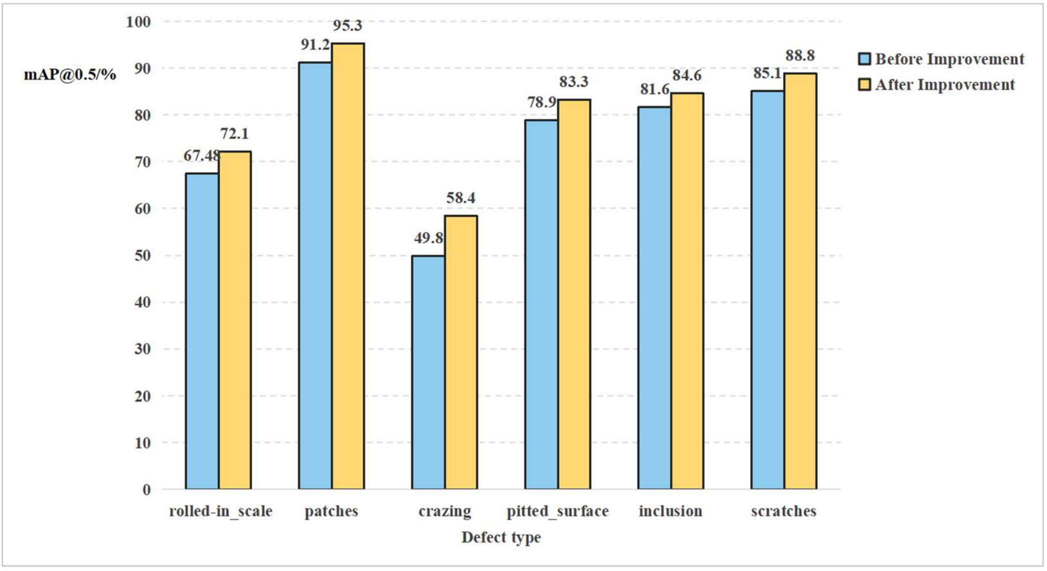

Figure 14. Comparison of average detection accuracy of various defects before and after improvement.

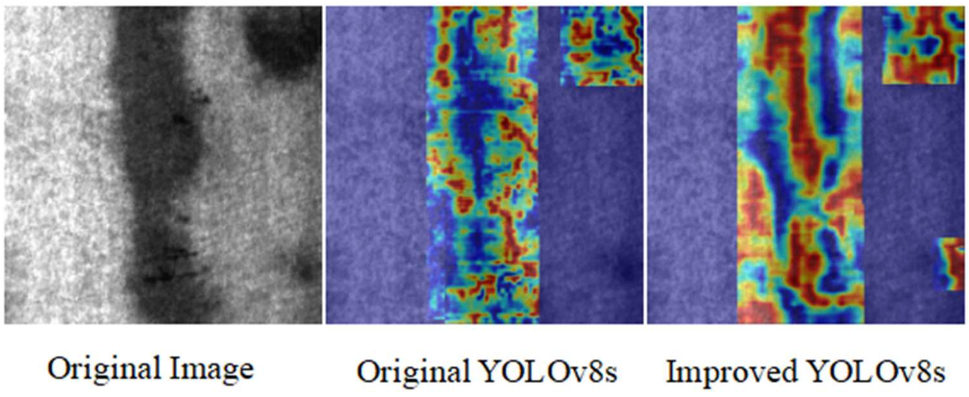

Figure 15. Comparison of heatmap results before and after the improvement of the YOLOv8 model.

4.5.2. Comparison with Other Algorithms

To further validate the detection performance of the improved algorithm, 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], along with classic YOLO series models (YOLOv5s and YOLOv6s). The experimental results are detailed in Table 3.

In terms of performance, the improved YOLOv8s model achieves the best balance between accuracy and 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. These results confirm the advantages of the proposed method in industrial defect detection scenarios, as it combines high accuracy with low resource consumption.

Table 3. Performance Comparison of Various Algorithms.

Method	mAP	FLOPs	Params	FPS
YOLOv5s	75.6	7.2	8.7	195
YOLOv6s	78.2	9.8	12.4	165
Faster R-CNN	76.4	41.5	134.2	22
SSD	68.9	26.8	35.4	125
EfficientDey-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. Compared with YOLOv5s, although the number of parameters has slightly increased, the mAP has improved by 4.8%, mainly due to the introduction 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. Conclusion

Addressing the challenges of designing lightweight networks and detecting small targets for hot-rolled strip steel surface defects, this paper proposes an optimized YOLOv8s model.. The findings reveal that, compared to the original YOLOv8s, the optimized version significantly enhances detection accuracy, reduces model size, and accelerates detection 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.

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*.
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.