International Journal of Environmental Sciences
ISSN: 2229-7359
Vol. 11 No. 21s, 2025
https://theaspd.com/index.php

Weld Seam Image Defect Detection Technology Based On Improved Yolo Algorithm

Shaoyi Hu[1], Saiful Bahri Bin Mohamed[1], Xin He[2,3]
[1]Faculty of Innovative Design and Technology,Universiti Sultan Zainal Abidin ,Terengganu, Kuala Terengganu 21300, Malaysia; hushaoyi0118@163.com (S.H.)
[2]QingGong College,North China University of Science and Technology, HeBei TangShan 064000,China
[3]School of Mechanical and Automotive Engineering, Guangxi University of Science and Technology, Liuzhou 545006, China; hex_1998@163.com (X.H.)
*Correspondence: saifulbahri@unisza.edu.my (S.M.)

Abstract: *In response to the problem of insufficient accuracy in current object detection algorithms for weld seam image defect detection, an improved model based on YOLOv8, called Sim-YOLOv8s, is proposed. First, the C2f module is improved by incorporating the SimAM module to enhance the overall performance of the model, reduce computational redundancy, and speed up network feature extraction. Second, the network structure is optimized based on the scale characteristics of weld seam defect targets to alleviate the loss of small target information caused by excessive downsampling. The Inner-CIoU is used as the new localization regression loss function to improve the learning capability for small target samples and accelerate the convergence of the bounding box regression. Finally, the model is pruned using Layer Adaptive Magnitude Pruning (LAMP), which sacrifices a certain amount of accuracy to reduce the model size and the number of parameters, enabling fast detection on embedded devices. Knowledge distillation is applied to compensate for the detection accuracy lost due to pruning, thereby improving the model's detection performance. Experimental results show that, with a threshold of 0.5, the Sim-YOLOv8s model achieves an average precision (mAP50) of 93.9%, 94.3%, and 97.2% for detecting pores, inclusions, and lack of fusion defects, respectively, improving by 2.4, 2.5 and 1.7 percentage points over the original model, demonstrating better weld seam defect detection performance.*

Keywords: *Image Processing; Object Detection; YOLOv8s; SimAM Module; Inner-CIoU Loss Function*

1. INTRODUCTION

Pipelines, as a long-distance and large-scale carrier for oil and gas transportation, play an important role in a country's resource transportation, economic development, and environmental protection. Therefore, ensuring pipeline safety is critical. Different-shaped components are connected to form a pipeline through welding. Still, due to process and technical limitations, weld seams may develop various types of surface defects, which affect pipeline safety [1].

For weld seam surface image detection, early detection methods often relied on manual inspection, which is costly, slow, and highly dependent on the experience level of the operator, making it unsuitable for modern detection needs [2]. In recent years, with the continuous advancement of deep learning technology, neural networks have been widely applied in the field of computer vision and have also been introduced into pipeline weld seam image processing[3-5].

Currently, mainstream object detection models are divided into two categories: single-stage and two-stage models. Two-stage object detection models first generate proposal boxes from the input image and then select the best detection results from these proposals. The advantage of this method is high detection accuracy, but the computational cost is significant, making it unsuitable for online monitoring [7].

Single-stage object detection models, on the other hand, directly predict the class and bounding boxes from the input image [8]. This method has lower computational requirements but does not achieve the same level of detection accuracy as the two-stage models. One typical single-stage object detection model is YOLO, which, due to its high real-time performance and ease of implementation using the PyTorch framework, has become one of the most popular algorithms in industrial inspection [9]. Many researchers have developed surface defect detection methods based on the YOLO framework [10-15].

It can be observed that most existing research on defect identification using CNNs faces issues such as complex model implementations and high computational costs, which makes it difficult to optimize model parameters and leads to longer training and inference times [16-18].

Guo et al. [19] proposed an MSFT-YOLO model based on YOLOv5 for steel surface defect detection. By introducing a transformer-based TRANS module, which integrates global information with local features, they improved the detection performance for defects with large variations in scale.Liu et al. [20] added a convolutional attention mechanism module to the convolutional layers of the YOLOv5 backbone network to guide the network to focus on key features. They also used the K-means clustering algorithm to reduce sensitivity to the initial clustering center, improving localization accuracy and reducing network loss [21].Shang et al. [22] proposed the RayNet network for defect feature extraction and classification. Sampath et al. [23] introduced the Defect Aux-Net framework based on multi-task learning and attention mechanisms to address defect recognition tasks [24].

However, in traditional deep learning-based defect recognition methods, few studies have focused on small defect sizes and low contrast between defects and backgrounds, which leads to insufficient detection accuracy for certain weld seam defects. This paper investigates from the perspective of lightweight networks and model compression, proposing an efficient lightweight welding defect detection algorithm called Sim-YOLO. The main contributions of this algorithm are as follows:

1. By introducing the Sim_C2f module to replace the original C2f module in the backbone network, the redundant computational load is effectively reduced, and memory access costs are minimized. The feature extraction capability of the model is preserved, while the computational speed is significantly improved.
2. The network structure is optimized based on the scale features of defect targets in the dataset, focusing on the detection of medium and small targets. This prevents the loss of small target information due to excessive downsampling and reduces the model's parameter size.
3. The Inner-SIoU is introduced as a new regression loss function, using auxiliary bounding boxes to accelerate the regression of low IoU samples and improve the localization accuracy for small targets.
4. The LAMP model pruning technique is used to reduce parameters and memory usage while maintaining as much accuracy as possible, decreasing the computational resource requirements. Knowledge distillation is applied to fine-tune the pruned model to enhance performance.

2. YOLOv8 Algorithm

YOLOv8 is an advanced object detection algorithm developed by Ultralytics. Its backbone network still uses the Darknet53 structure, which has been adopted since YOLOv3. In the architecture of YOLOv8, the C2f module is introduced, which deeply integrates the structural advantages of the C3 module in YOLOv5 and the Efficient Layer Aggregation Network (ELAN) module in YOLOv7. Specifically, the C2f module first performs a stacking operation on the BottleNeck module within the C3 module to enhance feature extraction capability. Then, features obtained through conventional convolution operations and detailed processing by three BottleNeck layers are concatenated according to specific rules, allowing multidimensional feature information to be aggregated. Finally, a convolutional layer is used to further refine the concatenated features, producing accurate and reliable final outputs, thus laying a solid foundation for YOLOv8’s outstanding performance in object detection tasks. This module can output richer gradient flow information while maintaining lightweight characteristics. Additionally, YOLOv8 adopts an anchor-free detection technique, discarding the Anchor Boxes used in previous versions for predicting the bounding boxes of target objects. This simplifies the detection process, reduces model complexity and computational cost, and also improves the flexibility and accuracy of detection. The proposal includes the YOLOv8 architecture diagram and the structural diagrams of its individual modules.

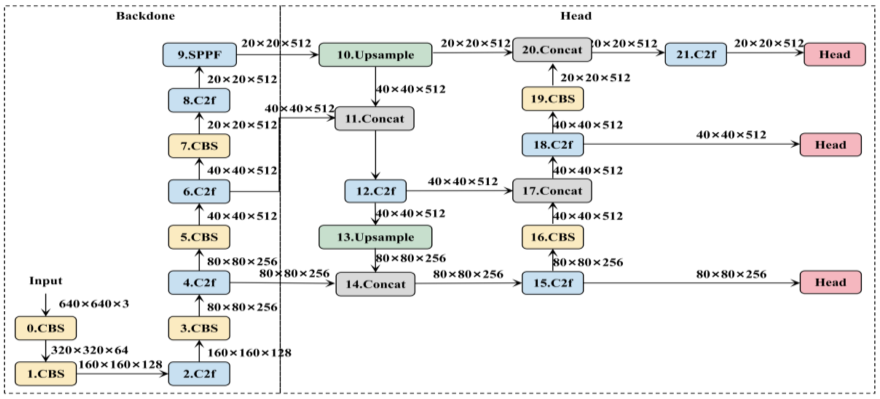


Figure 1. YOLOv8 network structure.

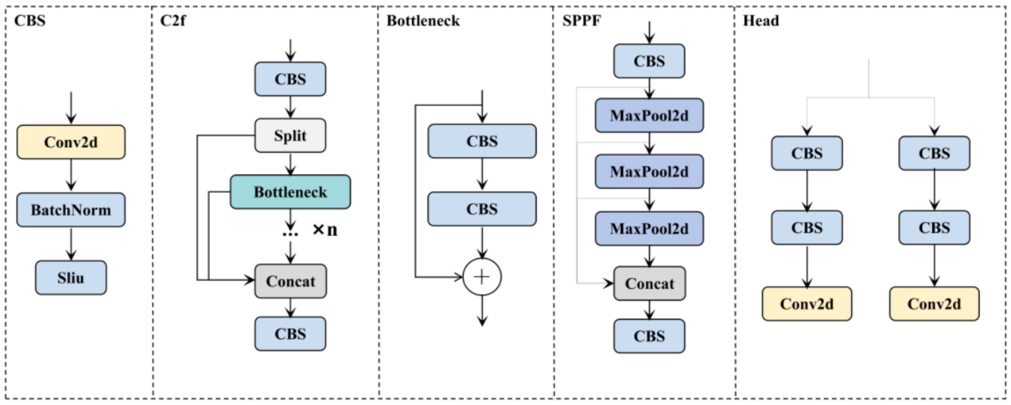


Figure 2. Structures of each module in YOLOv8.

YOLOv8 algorithm offers excellent flexibility, allowing researchers to select different versions of YOLOv8 based on the network depth and width. In this study, four versions of the detection performance were tested using a custom weld seam defect dataset, and the results are shown in Table 1. To save memory on embedded devices and reduce computational costs, YOLOv8s was chosen as the baseline model.

Table 1. Comparison of performance of different YOLO v8 models.

Model	Precision (%)	Recall (%)	Mean Average Precision (%)	Model Memory Usage (MB)
YOLO v8s	93.50	91.20	94.60	21.40
YOLO v8m	93.60	92.90	95.40	49.60
YOLO v8l	94.60	92.60	96.00	88.60
YOLOv8x	94.70	92.00	95.80	130.00

3. Algorithm Improvement

The structure of the improved YOLOv8 model (Sim-YOLOv8) is shown in Figure 3. First, the original C2f module in the network is replaced with the S_C2f module, which incorporates the SimAM module, to enhance the overall detection performance of the network. This improves detection accuracy and speed without affecting the model's parameter count and computational load. Next, the network structure is optimized to alleviate the issue of small target detail loss caused by excessively deep networks. Then, the loss function is changed to the Inner-CIoU loss function, which dynamically evaluates the quality of anchor boxes, making it easier to focus on anchor boxes of ordinary quality and improve their performance. Finally, the LAMP pruning technique is applied to reduce the number of parameters and memory usage while maintaining as much accuracy as possible, thus decreasing the computational resource requirements. Figure 3 shows the structure diagram of the improved model.

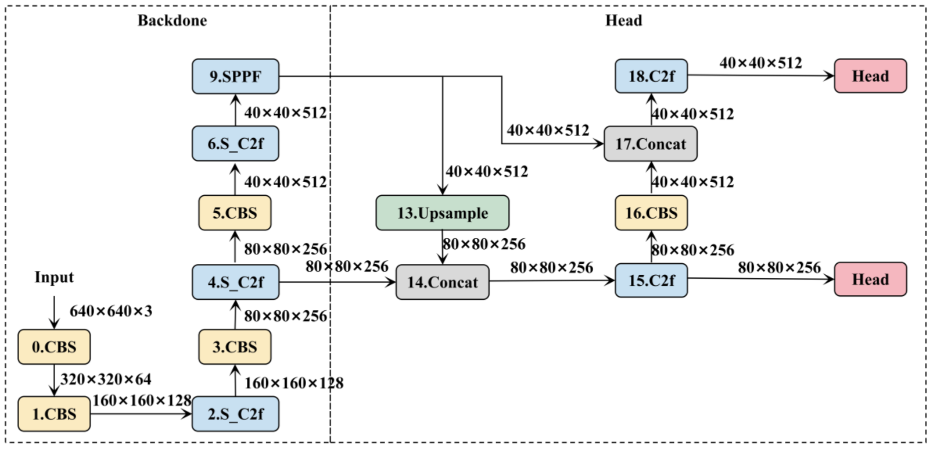


Figure 3. Sim_YOLO network structure.

3.1. Improved C2f

The C2f module, as a key component of the YOLOv8 network, integrates the advantages of the C3 module from YOLOv5 and the ELAN module from YOLOv7. It stacks the BottleNeck module from C3, concatenates the features from convolutional layers and the three BottleNeck layers, and outputs the result through a convolutional layer. This not only strengthens feature extraction, capturing multi-dimensional features, but also enhances both accuracy and speed, making YOLOv8 perform excellently. In the optimization process of the YOLOv8 network, to further exploit the potential of the C2f module, the focus is on enriching its gradient flow and enhancing its feature expression capability. Specifically, the SimAM module is introduced at the critical BottleNeck location. SimAM has a unique advantage: it can accurately compute the weight of each position, and notably, this addition does not impose additional computational burden on the model. By adding SimAM to the BottleNeck module in C2f, the feature representation ability of the module is improved. Figure 4 shows the improved S_C2f module.

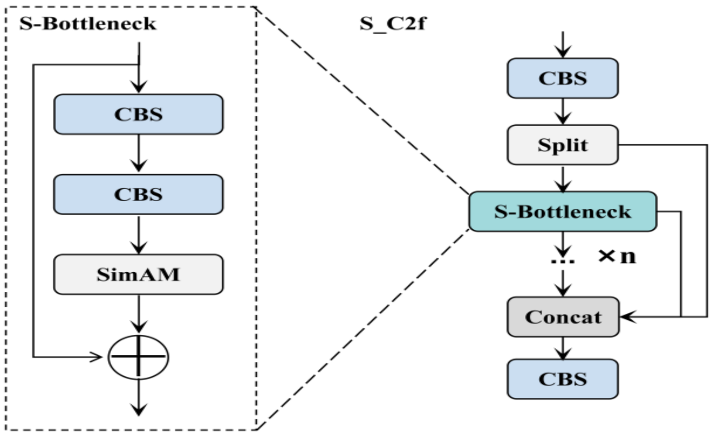


Figure 4. Structure diagrams of improved S-C2f module.

3.2. Improve Network Structure

The original YOLOv8 model has three prediction heads, corresponding to the detection of small, medium, and large targets, respectively. The large target detection layer performs 32x downsampling on the image. When the target size is smaller than 32 pixels, only one point of the target is sampled, which may be redundant when detecting small-scale defects in weld seams. Additionally, as the scale of the feature map decreases, the detailed information contained within it is lost, which is also detrimental to the detection of small-sized targets. To address these issues, the optimized architecture proposed in this paper is shown in Figure 5.

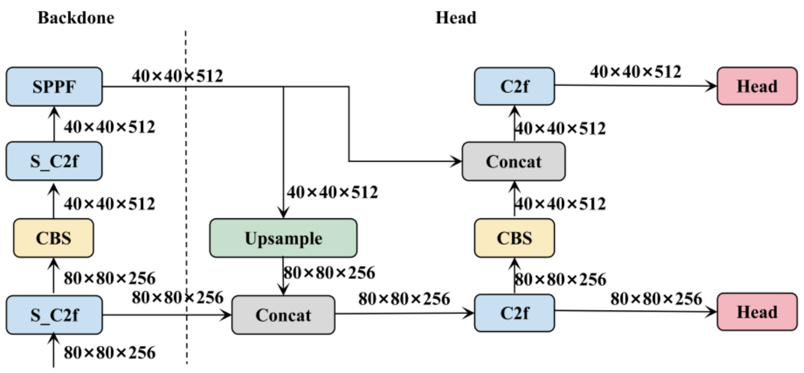


Figure 5. Network Structure Improvement.

The specific optimization strategy is as follows: In the initial phase, a CBS module and an S_C2f module at the end of the original backbone network are precisely removed. With this adjustment, the network performs at most 16x downsampling operations, generating a 40×40 feature map. This ensures that the detailed information of small targets is retained to the maximum extent during the network's forward propagation process, effectively preventing excessive information loss. Next, an upsampling operation is performed, and the resulting feature map is concatenated with the output from the second S_C2f module in the backbone network. After being processed by the C2f module, the concatenated feature map is passed to a detection head specifically designed for small target prediction. Then, only one downsampling step is carried out, ultimately obtaining a 40×40 output feature map, which will directly be applied to the detection of medium-sized defect targets. Through this optimized architecture, the issue of significant small target feature loss caused by frequent upsampling and downsampling in the network is successfully mitigated. While maintaining the accurate feature representation capability for small and medium targets in weld seam defect detection, this approach significantly reduces the consumption of computational resources.

3.3. Optimized Loss Function

Regression loss functions are an important method for measuring the accuracy of object detection. In YOLOv8, CIoU is used as the loss function, which considers three aspects: the overlap area between the predicted and ground truth boxes, the distance between their center points, and the aspect ratio. It is defined as follows:

$$L_{CIOU} = 1 - IoU + \frac{\rho^2(b, b^{gt})}{c^2} + \alpha v \tag{1}$$

α is a trade-off parameter:

$$\alpha = \frac{v}{(1 - IoU) + v} \tag{2}$$

v measures the consistency of the aspect ratio:

$$v = \frac{4}{\pi^2}\left(\arctan\frac{w^{gt}}{h^{gt}} - \arctan\frac{w}{h}\right)^2 \tag{3}$$

w^{gt} and h^{gt} represent the width and height of the ground truth bounding box, while w and h represent the width and height of the predicted bounding box.

Although CloU can effectively improve detection performance by adding new geometric constraints, it cannot effectively address the issue of multiple damage samples present in the dataset scene. Therefore, this paper further introduces the concept of Inner-IoU, which generates auxiliary bounding boxes to calculate IoU loss, improving the model's ability to localize small samples and accelerating the regression of predicted bounding boxes. The definition of Inner-CloU is as follows:

$$L_{Inner-CIOU} = 1 - IoU_{inner} + \frac{\rho^2(b, b^{gt})}{c^2} + \alpha v \tag{4}$$

The ground truth box (GT Box) and the predicted box (Pred Box) are represented as b^{gt} and b, respectively. (x_c^{gt},y_c^{gt}) represents the center point of the GT Box and the Inner GT Box. (x_c,y_c) represents the center point of the Pred Box and the Inner Pred Box. The width and height of the GT Box are denoted as w^{gt} and h^{gt}, while the width and height of the Pred Box are represented as w and h. The variable ratio corresponds to the scaling factor, which typically ranges from 0.5 to 1.5, with a value of 1.2 used in this paper.

$$\begin{cases} b_l^{gt} = x_c^{gt} - \frac{w^{gt} \times ratio}{2}, b_r^{gt} = x_c^{gt} + \frac{w^{gt} \times ratio}{2} \\ b_t^{gt} = y_c^{gt} - \frac{h^{gt} \times ratio}{2}, b_b^{gt} = y_b^{gt} + \frac{h^{gt} \times ratio}{2} \\ b_l = x_c - \frac{w \times ratio}{2}, b_r = x_c + \frac{w \times ratio}{2} \\ t = y_c - \frac{h \times ratio}{2}, b_b = y_c + \frac{h \times ratio}{2} \end{cases} \tag{5}$$

$$inter = \left(\min(b_r^{gt}, b_r) - \max(b_l^{gt}, b_l)\right) \times \left(\min(b_b^{gt}, b_b) - \max(b_t^{gt}, b_t)\right) \tag{6}$$

$$union = (w^{gt} \times h^{gt}) \times (ratio)^2 + (w \times h) \times (ratio)^2 - inter \tag{7}$$

$$IoU^{inner} = \frac{inter}{union} \tag{8}$$

3.4. Model Pruning and Knowledge Distillation

Deploying neural networks on embedded devices is challenging due to the limited storage space and computational resources, requiring model compression to reduce computational load and parameter count. Neural networks often contain redundant parameters, which can be pruned using the LAMP (Layer-adaptive sparsity for magnitude-based pruning) technique. The approach is as follows:

In a neural network, weight tensors can either be in the form of a 2D matrix or a 4D tensor. To unify the definition of the LAMP score, all weight tensors need to be unfolded into one-dimensional vectors. For each unfolded vector, the elements are sorted according to the given index. Let μ and v be the weight indices, and $W[\mu]$ and $W[v]$ be the corresponding mapped weights, with the condition that when $\mu < v$, $W[\mu] \leq W[v]$. The LAMP score calculation formula is as follows:

$$\mathrm{sorce}(\mu, W) = \frac{(W[\mu])^2}{\sum_{v\geq\mu}(W[v])^2} \tag{9}$$

$(W[\mu])^2$ represents the squared weights of the target connection, and $\sum_{v\geq\mu}(W[v])^2$ represents the sum of the squared weights of the connections with indices greater than μ within the same layer. That is, the denominator represents the sum of squared weights of all remaining connections in the current layer, while the connections with indices smaller than μ have already been pruned. From equation (9), it can be seen that the larger the squared weight, the higher the LAMP score. This score reflects the importance of the connection, and weights with lower scores are considered unimportant and are pruned. During LAMP pruning, after setting the target pruning ratio, low-score connections are globally pruned to meet the global sparsity constraint, with at least one connection in each layer having a score of 1 to prevent layer collapse. The application of LAMP can reduce parameters and is better suited for embedded devices.
Knowledge distillation is an effective method to further improve the model's detection accuracy. For the pruned model, distillation techniques are further applied to enhance its performance. First, based on the results from the damage dataset, a teacher model is carefully selected. To ensure that both the teacher and student models extract feature maps at the same scale, the improved Sim_YOLOv8s model is chosen as the teacher. A distillation method suitable for dense object detection and cross-task consistency protocols is used. This method transfers knowledge by mimicking the teacher detector's logit output, which includes binary classification distillation loss L_{cls}^{dis} and IoU-based localization distillation loss L_{los}^{dis}, thereby transferring classification and localization knowledge. The process is shown in Figure 6.

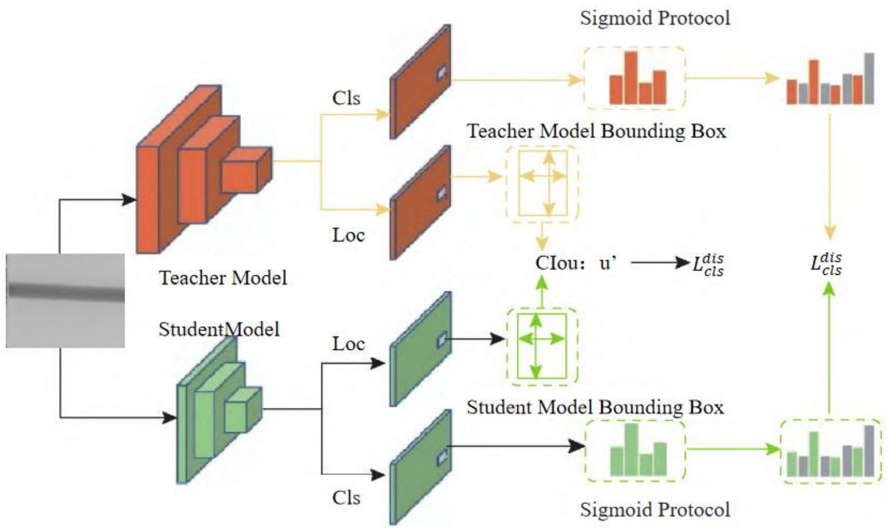


Figure 6. Knowledge distillation flowchart.
The distillation process for the dataset input is divided into two parts: (1)Input the defect damage data into the Sim_YOLO teacher model. (2)Input the defect damage data into the pruned student model and obtain the binary classification maps from both the teacher and student models.
The binary cross-entropy loss values for each binary classification map are computed to extract classification knowledge. The importance of the classification maps is then weighted, yielding the binary distillation loss A, which enables the student model to inherit the teacher model's correct prediction capabilities. The formula for calculating the binary classification distillation loss is as follows:

$$L_{cls}^{dis}(x) = \sum_{i=1}^{n}\sum_{j=1}^{n} w_{i,j} L_{BCE}\left(p_{i,j}^{t'}, p_{i,j}^{x'}\right) \tag{10}$$

$$L_{BCE}\left(p_{i,j}^{t'}, p_{i,j}^{x'}\right) = -\left[\left(1 - p_{i,j}^{t'}\right) \times \lg\left(1 - p_{i,j}^{x'}\right) + p_{i,j}^{t'} \times \lg p_{i,j}^{x'}\right] \tag{11}$$

$w_{i,j}$ represents the importance weight of sample x; L_{BCE} is the binary cross-entropy loss value; $p_{i,j}^{t'}$、$p_{i,j}^{x'}$ represent the binary classification scores at the i-th position and j-th class, respectively.
Similarly, along with obtaining the binary classification maps, the teacher model's and student model's localization maps are also obtained. The Intersection over Union (IoU) between the teacher model's

bounding boxes and the student model's bounding boxes is calculated. A loss weighting strategy is then introduced to obtain the IoU-based localization distillation loss L_{los}^{dis}, which transfers localization knowledge from the teacher model to the student model. The formula for calculating the localization distillation loss is as follows:

$$L_{los}^{dis}(x) = \sum_{j=1}^{n} \max(w, j) \times (1 - u_i') \tag{12}$$

u_i' represents the IoU between the teacher model's bounding boxes and the student model's bounding boxes.

The combined distillation loss is calculated as follows:

$$L_{total}^{dis}(x) = a_1 L_{cls}^{dis}(x) + a_2 L_{los}^{dis}(x) \tag{13}$$

a_1 and a_2 is the distillation weight coefficient.

4. Experiment and Result Analysis

4.1. Experimental Platform

To verify the effectiveness of the improved algorithm under different experimental environments, three different experimental platforms were set up, including a computer development platform and two embedded test platforms.

The computer development platform uses the Windows 10 x64 operating system, with an Intel® Core™ i5-11400 CPU, NVIDIA Geforce RTX-3060 12GB GPU, and 32 GB RAM.

The embedded test platform utilizes the Raspberry Pi 4B device developed and manufactured by Raspberry Pi Holdings plc, with an ARM Cortex-A72 1.5GHz (quad-core) CPU, and 4GB LPDDR4 RAM. This platform is primarily used for model inference and testing. The experimental environment runs on the Ubuntu 22.04 system, with Python 3.11 as the programming language and Pytorch 1.10.2 as the deep learning framework.

4.2. Model Evaluation Metrics

This study uses Precision (P), Recall (R), and Mean Average Precision (mAP) as evaluation metrics for the algorithm's detection performance. The calculation formulas are as follows:

$$P = \frac{TP}{TP + FP} \tag{14}$$

$$R = \frac{TP}{TP + FN} \tag{15}$$

$$AP = \int_0^1 P(R)dR \tag{16}$$

$$mAP = \frac{\sum_{i=1}^{n} AP_i}{n} \tag{17}$$

Where FP represents the number of incorrectly predicted negative samples, FN represents the number of incorrectly predicted positive samples, and TP represents the number of correctly predicted positive samples. AP reflects the accuracy of target detection for a single class, and mAP represents the average AP value over multiple classes.

4.3. Model Pruning Experiment

During the model pruning process, a significant phenomenon was observed: as the compression ratio gradually increased, the model's loss in accuracy rose correspondingly. The detailed data is presented in Table 2, which shows the model's performance metrics under different compression ratios. When the compression ratio is below 3, the negative impact on precision, recall, and mean average precision is relatively minor, while the floating-point operations, parameter count, and memory usage are significantly reduced, allowing more room for model optimization. However, once the compression ratio exceeds the threshold of 3, the situation takes a sharp turn, with recall and mean average precision showing a dramatic decline. Despite this, the changes in parameter count and memory usage are not significant. This phenomenon clearly indicates that when the compression ratio exceeds 3, pruning the channels starts to interfere with the core performance of the model. Therefore, this study ultimately selects a pruning strategy with a compression ratio of 3, using it as a baseline for precise pruning, aiming to achieve efficient resource utilization while maintaining model performance.

Table 2. Comparison of Detection Performance Across Different Networks.

Compression Ratio	P (%)	R (%)	mAP50 (%)	FLOPs	Params	Model Memory Usage (MB)
2.0	94.8	91.6	95.3	$9.6*10^9$	$6.799*10^6$	11.01
2.5	93.7	91.2	95.1	$9.3*10^9$	$5.493*10^6$	10.49
3.0	93.5	90.7	94.8	$8.2*10^9$	$4.134*10^6$	9.64
3.5	91.6	89.7	92.3	$7.4*10^9$	$3.833*10^6$	8.16

4.4. Knowledge Distillation Experimen

In the knowledge distillation experiment, it was observed that the distillation weight coefficient plays a crucial role in the distillation effectiveness. Based on the student model and teacher model (Sim-YOLOv8s), the impact of various distillation weight coefficients on the knowledge distillation process was explored in detail, as shown in Figure 7. From the figure, it can be seen that when the distillation weight coefficient is set to 2.0, the distilled model demonstrates a high precision (93.70%) and mean average precision (95.20%). As the distillation weight coefficient changes dynamically, the precision and recall curves exhibit fluctuations, indicating that different weight coefficients cause the model to focus on specific feature information of passion fruit. However, the mean average precision consistently remains at a high level, strongly supporting that the distilled model exhibits good performance in passion fruit detection.

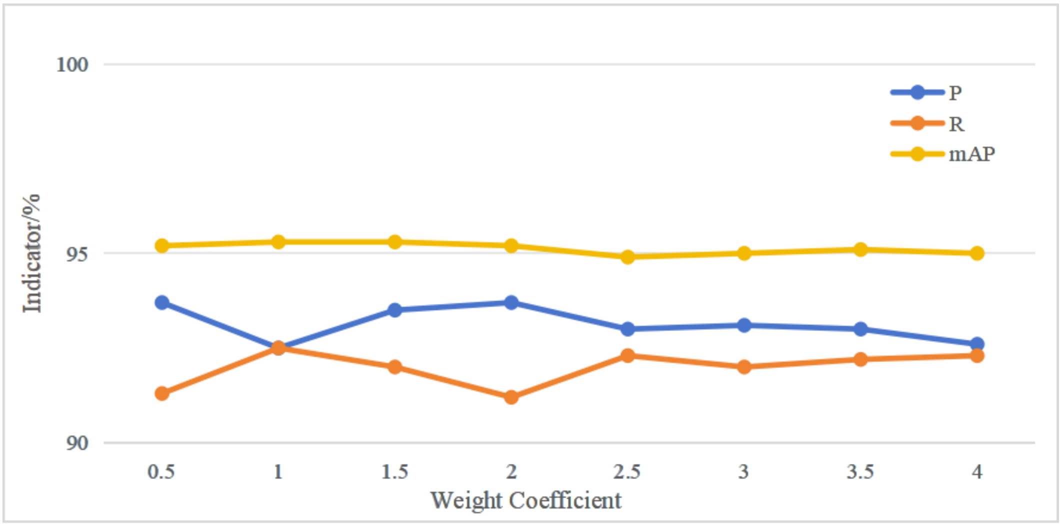


Figure 7. Model Performance Under Different Weight Coefficients.

4.5. Ablation Experiment

Ablation experiments aim to analyze the value of each improvement. Based on the self-made dataset, ablation experiments were conducted to test the improved model. The experimental results are shown in Table 3.

Table 3. Comparison of Detection Performance Across Different Networks.

Group	Neck Network	Inner-CIoU	Pruning and Knowledge Distillation	P (%)	R (%)	mAP50 (%)	Params	Model Memory Usage (MB)
0	×	×	×	86.7	84.9	88.6	$1.236*10^7$	20.60

1	√	×	×	91.9	91.1	93.1	$1.594*10^{7}$	25.60
2	√	√	×	93.5	90.7	94.8	$4.134*10^{6}$	18.95
3	√	√	√	94.7	91.3	95.2	$4.134*10^{6}$	9.64

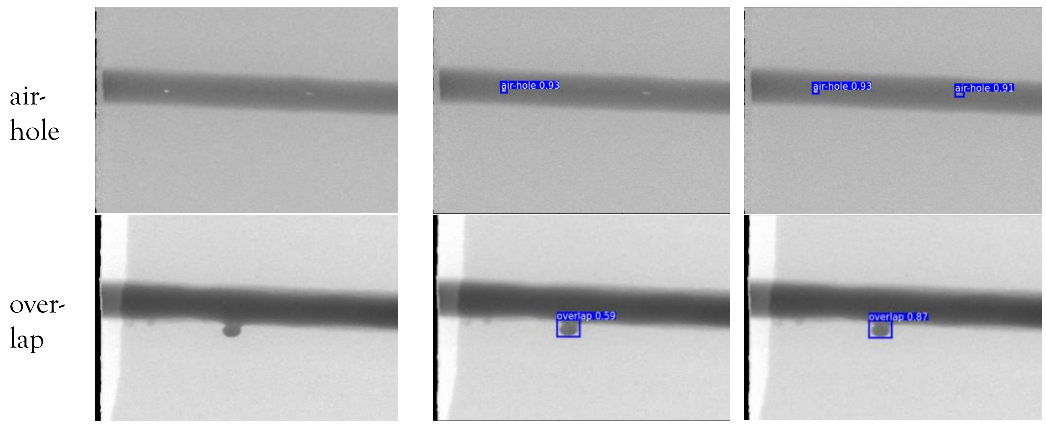
The results from the table show that Group 1 introduced SimAM and improved the C2f module. After adding SimAM and optimizing the network structure, the model's parameter count and floating-point operations remained unchanged, but both accuracy and detection speed were improved. Group 2 built upon Group 1 by changing the loss function to Inner-CIoU, which slightly reduced the detection speed but increased the model's mAP50 by 1.7 percentage points compared to the original model. Group 3 further enhanced Group 2 with model pruning and knowledge distillation, resulting in no loss of accuracy, while the model's parameter count and memory usage were reduced by 66.55% and 53.20%, respectively, making it more suitable for embedded terminal deployment.

4.6. Comparison of Different Models

In order to compare the performance of the improved model with current mainstream object detection models, the experiment uses metrics such as FLOPs, parameter count, frame rate, precision, recall, mean average precision (mAP), and model memory usage. The proposed model is compared with Faster R-CNN, YOLO v5s, YOLO v7 Tiny, YOLO v8s, and YOLO v9 on the same self-constructed dataset. The experimental results are shown in Table 4 and Figure 8.

Table 4. Comparison of Detection Performance Across Different Networks.

Model	AP/%			mAP50(%)	FPS (computer)	FPS (4B)	Params	FLOPs
	Pore	Slag inclusion	Incomplete penetration					
Sim-YOLOv8s	93.9	94.3	97.2	95.2	98.30	5.86	$4.134*10^{6}$	$8.196*10^{9}$
YOLO v9	91.2	92.3	94.8	92.9	49.68	1.10	$3.148*10^{7}$	$1.654*10^{11}$
YOLO v8s	91.5	91.8	95.5	93.6	84.62	3.00	$1.236*10^{7}$	$2.679*10^{10}$
YOLOv7 Tiny	86.2	89.4	93.4	86.8	56.45	1.32	$3.981*10^{7}$	$1.195*10^{11}$
YOLO v5s	88.9	90.4	76.2	84.3	71.60	4.10	$7.034*10^{6}$	$1.764*10^{10}$
Faster R-CNN	75.3	84.6	37.5	57.9	6.01	0.23	$4.128*10^{7}$	$3.145*10^{11}$



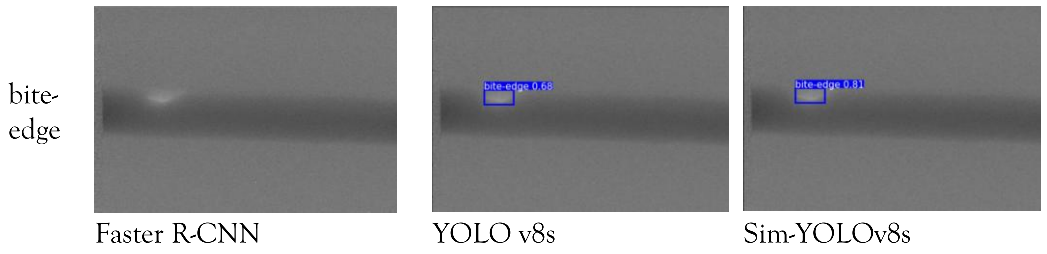


Figure 8. Comparison of detection performance among three models.

In the comparison, Sim-YOLOv8 demonstrates exceptional performance. Compared to models like YOLOv9, YOLOv8s, YOLOv7 Tiny, YOLO v5s, and Faster R-CNN, Sim-YOLOv8 shows significant advantages in various parameters, with its accuracy metrics leading the way. The further optimized Sim-YOLOv8s model, compared to YOLOv8s, achieves breakthroughs in multiple aspects: on one hand, it reduces the requirements for Params and FLOPs; on the other hand, in defect detection performance, the average precision for detecting porosity, slag inclusion, and lack of fusion defects improved by 2.4, 2.5, and 1.7 percentage points, respectively, and the mAP increased by 1.6 percentage points.

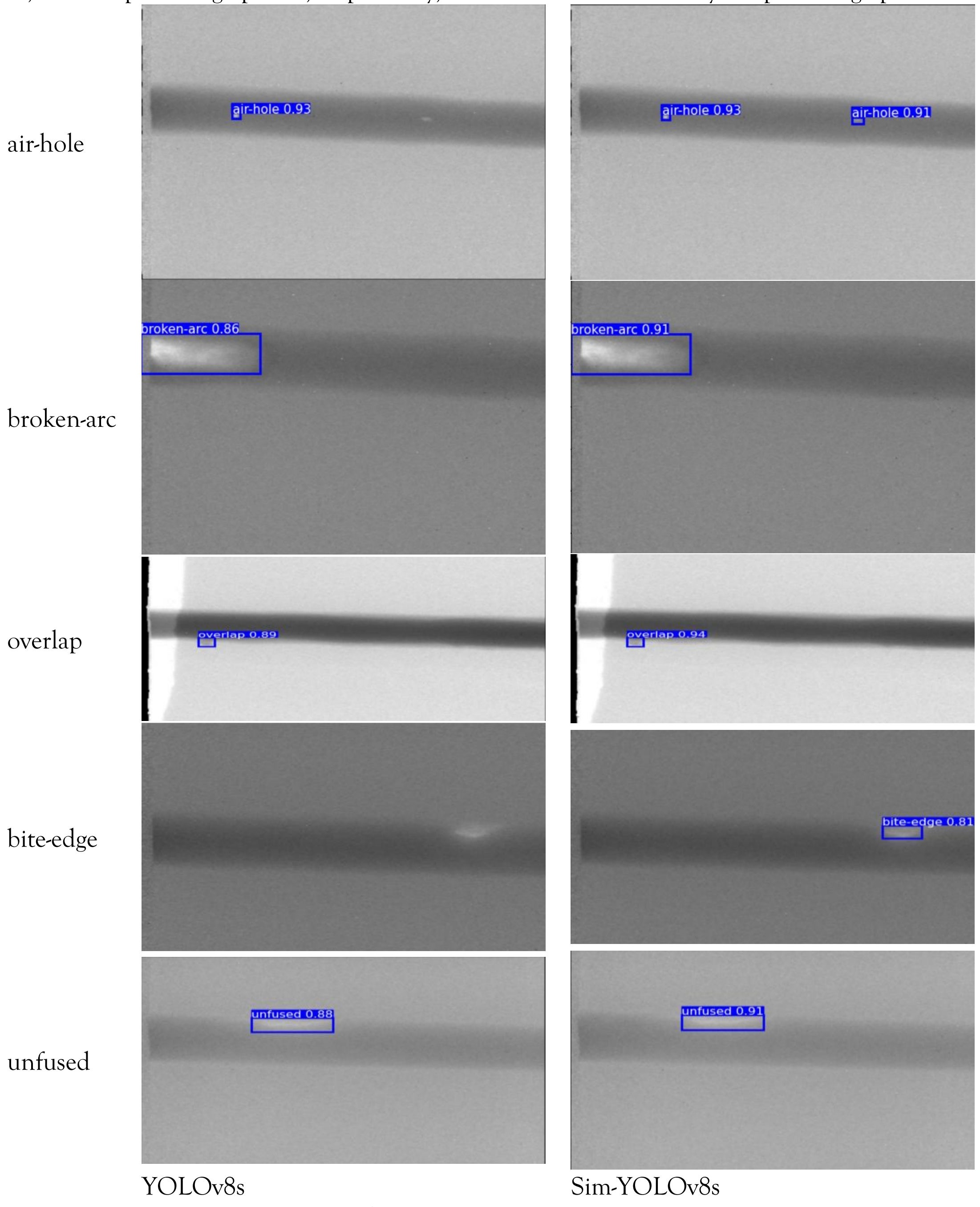


Figure 9. Comparison Chart of Model Improvement Effects.

5. CONCLUSION

A pipeline weld seam image defect detection model based on the improved model Sim-YOLOv8 is proposed. By comparing it with the original YOLOv8 model and current mainstream models, the superiority of the improved model in detecting pipeline weld seam image defects is validated. The comparison metrics include mAP50, FPS, parameter count, and floating-point operations. The main conclusions are as follows:

(1) The original network structure was adjusted by removing the detection head for large targets and redundant network layers, which reduced the network parameters while improving the detection performance.

(2) A series of effective improvements were made. On one hand, SimAM was introduced to enhance the C2f module, significantly improving detection accuracy and speed without changing the model's computational load. On the other hand, the newly added Inner-CIoU loss function improved the accuracy of detecting anchor boxes.

(3) The results show that the improved lightweight model outperforms YOLO v5s and YOLO v8s, with better performance in weld seam defect recognition. It can quickly and accurately detect weld defects in complex environments and multi-target scenarios, balancing both accuracy and real-time performance. This demonstrates that the improved model has optimal overall performance, robust capabilities, and can effectively support weld seam defect recognition.

Author Contributions: Methodology, S.M.; software, S.H.; validation, S.H.; resources, S.H. and X.H; data curation, X.H.; writing—review and editing, S.H.; funding acquisition, S.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Central Government-Guided Local Science and Technology Development Fund Program, grant number 2023PRJ0101, Liuzhou Science and Technology Program Project, grant number 2022AAB0101.

Institutional Review Board Statement: Not applicable

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

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

REFERENCES

1.Smith, J.; Lee, R.; Wang, Q.; et al. Advances in Pipeline Welding Defect Detection: A Review. J. Pip. Eng. **2021**, 45, 123–135.
2.Brown, T.; Zhao, P.; Yang, M. A Survey on Surface Defect Detection Methods. Int. J. Comput. Vis. **2020**, 88, 24–36.
3.Zhang, F.; Liu, D.; Sun, H.; et al. Deep Learning for Defect Detection in Pipeline Welding: A Review. Appl. Sci. **2020**, 10, 1001.
4.Zhang, X.; Wang, S.; Li, J. Convolutional Neural Networks for Surface Defect Detection in Steel Manufacturing. J. Manuf. Process. **2020**, 58, 292–304.
5.Liu, Y.; Li, C.; Zhang, W.; et al. Intelligent Welding Seam Defect Detection Using Convolutional Neural Networks. J. Robot. Comput.-Integr. Manuf. **2021**, 69, 29–41.
6.Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Trans. Pattern Anal. Mach. Intell. **2015**, 39, 1137–1149.
7.Liu, W.; Anguelov, D.; Erhan, D.; et al. SSD: Single Shot MultiBox Detector. Proceedings of the European Conference on Computer Vision (ECCV), 2016, 21–37.
8.Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2016, 779–788.
9.Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. arXiv 1804.02767 **2018**.
10. Li, H.; Zhang, X.; He, Z.; et al. Surface Defect Detection Based on YOLOv5 for Industrial Steel Production Lines. Sensors **2021**, 21, 3456.
11. Chen, C.; Yu, Z.; Zhang, Y.; et al. An Improved YOLOv5 for Surface Defect Detection in Automated Manufacturing. Int. J. Adv. Manuf. Technol. **2022**, 116, 2107–2118.
12. Yang, L.; Wu, X.; Luo, J.; et al. A Deep Learning-Based Approach for Real-Time Defect Detection in Industrial Welds Using YOLO. Comput. Ind. Eng. **2020**, 149, 106750.
13. Li, Y.; Guo, Y.; Yang, S.; et al. Defect Detection in Pipeline Welds Using YOLOv4 and Feature Fusion. Appl. Sci. **2021**, 11, 2712.
14. Zhao, H.; Zhang, W.; Lin, L.; et al. Enhancing Weld Defect Detection Using YOLOv4 and Attention Mechanisms. J. Manuf. Sci. Eng. **2021**, 143, 101–110.
15. Liu, X.; Zhang, T.; Yang, H.; et al. Steel Surface Defect Detection Using YOLO-based Model. J. Comput. Sci. Technol. **2020**, 35, 1017–1027.
16. Li, X.; Zhang, J.; Liu, Y.; et al. Challenges in Industrial Surface Defect Detection Using Convolutional Neural Networks. IEEE Trans. Ind. Inform. **2020**, 16, 4826–4835.

17. Li, M.; Xu, Y.; He, L.; et al. Research on the Automatic Detection of Weld Surface Defects Using Deep Learning Algorithms. Int. J. Adv. Robot. Syst. **2019**, 16, 156.
18. Zhang, M.; Xu, Z.; Zhang, Q.; et al. Automatic Detection of Weld Surface Defects Using YOLOv3-based Approach. Appl. Opt. **2019**, 58, 7455–7463.
19. Guo, Z.; Zhang, H.; He, Y.; et al. MSFT-YOLO: Improved YOLOv5 Based on Transformer for Detecting Defects of Steel Surface. Sensors **2022**, 22, 3467.
20. Guo, Z.; Zhang, Z.; Liu, Y.; Li, X.; Wang, Z.; Liu, S. MSFT-YOLO: Improved YOLOv5 Based on Transformer for Detecting Defects of Steel Surface. Sensors **2022**, 22, 3467.
21. Liu, X.Y.; Peng, H.W.; Zheng, N.X.; et al. EfficientViT: Memory Efficient Vision Transformer with Cascaded Group Attention. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023; pp. 14420–14430.
22. Shang, J.; Li, X.; Zhao, M.; Zhang, Z.; Wang, Y.; Chen, H. Oil Pipeline Weld Defect Identification System Based on Convolutional Neural Network. KSII Trans. Internet Inf. Syst. **2020**, 14, 1086–1103.
23. Sampath, V.; Aravind, P.; Singh, R.; Bhattacharya, S.; Bhat, N.; Thakur, P.; Gupta, A. Attention-Guided Multitask Learning for Surface Defect Identification. IEEE Trans. Ind. Inform. **2023**, 19, 9713–9721.
24. Zhao, Y.; Zhang, Y.; Li, H.; et al. A Lightweight Convolutional Neural Network for Real-Time Surface Defect Detection. J. Electron. Imaging **2022**, 31, 023003.