International Journal of Engineering Inventions
e-ISSN: 2278-7461, p-ISSN: 2319-6491
Volume 14, Issue 11 [November 2025] PP: 20-30

Design and Implementation of a Casting Defect Detection System Based on YOLOv8

Wenqi Sun[1], Zhida Wu[1], Xiaojia Shi[1], Yingkai Shen[1], Jiayu Chen[1] ,Yuelang Tang[1], Rui Cao[1*]

[1]*College of Mechanical and Vehicular Engineering, Changchun University, 130022, Changchun, China*
**Corresponding Author: Rui Cao*

Abstract: *Aiming at the problems in industrial casting defect detection, such as difficulty in identifying micro-defects, interference from the coexistence of multi-category defects, and strict real-time requirements, a casting defect detection system based on YOLOv8 is proposed. The system uses CSPDarknet53 as the backbone network, optimizes the network structure by introducing depthwise separable convolution and Ghost module, adopts the Task-Aligned Assigner dynamic label assignment strategy, integrates the SPPF multi-scale feature pooling module, and abandons the traditional anchor box design to adopt an Anchor-free mechanism. Experimental results on the industrial casting defect dataset show that the detection speed of the system reaches 120 FPS (in the Tesla T4 GPU environment), the single-frame detection time is ≤ 50 ms after acceleration by TensorRT, and the mAP@0.5 exceeds 90%. Compared with traditional machine vision methods, the missed detection rate is reduced to less than 5%, and only 300-500 defect samples are needed to complete model fine-tuning. This system effectively solves the problems of low efficiency, poor robustness, and insufficient scalability of traditional detection methods, and meets the requirements of real-time and high-precision detection in industrial production lines.*

Keywords: *Casting defect detection; YOLOv8; Depthwise separable convolution; Ghost module; SPPF multi-scale feature pooling module; Anchor-free mechanism; Feature fusion; Real-time detection*

Date of Submission: 25-10-2025 Date of acceptance: 05-11-2025

I. INTRODUCTION

As core fundamental components in the machinery manufacturing industry, castings are widely used in fields such as automotive, aerospace, and construction machinery. Their surface and internal defects (e.g., blowholes, cracks, inclusions, depressions, etc.) directly affect the service safety and service life of products [1]. In industrial production, casting defect detection faces three core challenges: first, defects are extremely small (on the millimeter scale), making localization difficult against complex backgrounds; second, multiple types of defects often coexist, requiring high fine-grained recognition capabilities from the model; third, production lines demand millisecond-level detection speed while maintaining an accuracy rate of over 99% [2].

Traditional casting defect detection methods mainly rely on manual visual inspection and conventional machine vision technology. Manual visual inspection depends on the experience of inspectors, with a detection speed of only 3–5 minutes per piece; prolonged work easily leads to visual fatigue, resulting in a missed detection rate as high as 15%–20% [3]. Conventional machine vision algorithms based on threshold segmentation and edge detection are sensitive to interferences such as lighting changes and surface reflections, and have poor generalization ability. Moreover, for new types of defects, feature extraction rules need to be redesigned, leading to a development cycle of 2–3 months[4].

In recent years, deep learning-driven object detection technology has provided a new approach for industrial defect detection. The YOLO series algorithms, leveraging the advantage of end-to-end detection, achieve a good balance between detection speed and accuracy [5]. As one of the latest versions in this series, YOLOv8 possesses stronger feature extraction capabilities and more flexible task adaptability. Based on the YOLOv8 model, this paper conducts targeted optimizations for the casting defect detection scenario. Through improvements such as lightweight network design, dynamic sample assignment, and multi-scale feature fusion, it realizes accurate recognition of micro-defects, simultaneous detection of multi-category defects, and real-time inference, thereby providing an efficient solution for quality control of industrial castings.

II. Improved Design of the YOLOv8 Model

2.1 Original Core Architecture of YOLOv8

YOLOv8 adopts CSPDarknet53 as its backbone network, and reduces computational complexity while enhancing feature extraction capabilities through the Cross-Stage Partial (CSP) structure [6]. Its core architecture consists of three parts: Backbone (feature extraction), Neck (feature fusion), and Head (detection output). It supports the integrated processing of object detection, instance segmentation, and classification tasks, and the model output is compatible with ONNX and TensorRT formats, facilitating industrial-grade deployment.

2.2 Lightweight Optimization of the Backbone Network

The original Backbone of YOLOv8 adopts standard convolution and the CSPDarknet53 architecture. Although it possesses strong feature extraction capabilities, it has a large number of parameters and high computational complexity (the original YOLOv8s has approximately 21.9 million parameters and around 28.4 GFLOPs), making it difficult to adapt to the deployment requirements of edge computing devices (such as embedded GPUs and FPGAs) in industrial scenarios.

To address this issue, this study implements network lightweighting while maintaining unchanged defect feature extraction capabilities through two core modifications: replacing standard convolution with Depthwise Separable Convolution and integrating the Ghost module.

2.2.1 Depthwise Separable Convolution

First, Depthwise Separable Convolution decomposes standard convolution into depthwise convolution and pointwise convolution, which respectively complete spatial feature extraction and channel fusion. On the premise of maintaining unchanged feature extraction capabilities, it reduces the number of model parameters and computational load [7]. Their respective roles are as follows:

Depthwise convolution: An independent convolution kernel is assigned to each input channel to extract only spatial features. At this point, the computational load is [8]:

$$F_{dw} = H \times W \times C_{in} \times K \times K \quad (1)$$

Among them, $H \times W \times C_{in}$ represents the size of the input feature map (height × width × number of input channels), and $K \times K$ is the size of the convolution kernel.

Pointwise convolution: 1×1 convolution is used to fuse channel features, and the computational load at this point is:

$$F_{pw} = 1 \times 1 \times C_{in} \times C_{out} \times H \times W \quad (2)$$

Among them, C_{out} represents the number of output channels.

At this point, the total computational load is:

$$F_{ds} = F_{dw} + F_{pw} \quad (3)$$

Whereas the computational load of standard convolution is:

$$F_{std} = K \times K \times C_{in} \times C_{out} \times H \times W \quad (4)$$

The ratio of the computational load between Depthwise Separable Convolution and standard convolution is:

$$\text{Compression Ratio} = \frac{F_{ds}}{F_{std}} = \frac{1}{C_{out}} + \frac{1}{K^2} \quad (5)$$

When k=3 (a commonly used convolution kernel size) and C_{out}=64, the compression ratio is approximately 12.3%, meaning the computational load is only about 1/8 of that of standard convolution.

2.2.2 Ghost Module

However,there are a large number of "redundant features" (i.e., similar feature maps) in the feature maps generated by traditional convolution. The Ghost module generates rich feature maps at low cost through the method of "base convolution generating core features" + "'simple linear transformation generating redundant features (Ghost features)", further enhancing the feature expression ability oflightweight networks.

The feature generation process of the Ghost module is divided into two steps:

- Core Feature Generation

First let the input feature map be $X \in R^{H\times W\times C_{in}}$,and then generate the core feature map $Y = Conv(X; W_1)$ using a small number of convolution kernels.

Among them $W_1 \in R^{K\times K\times C_{in}\times C_{mid}}$is the core convolution keel， $Y \in R^{H\times W\times C_{mid}}$($C_{mid} = \frac{C_{out}}{s}$,S is the redundancy coefficient,with a value range of 2-4), and Conv represents the standard convolution operation.

- Ghost Feature Generation

Perform a linear transformation (e.g., 3 X 3 convolution) on each channel of the core feature map Y to generate (s-1) similar feature maps, and finally obtainCout=S X Cmid feature maps through concatenation.

The processing performed here is [9]:

$$Y_i' = Linear(Y_i; W_i) \quad (i = 1,2,\ldots, C_{mid}) \tag{6}$$

$$Z = Concat(Y, Y_1', Y_2', \ldots, Y_{C_{mid}}') \tag{7}$$

Compared with the standard convolution that generates the same number of feature maps, the computational load compression ratio of the Ghost module is 1/s (with a 50% compression when s= 2). Moreover, through the enhancement of redundant features, the robustness of defect features is significantly improved.

2.2.3Structure of Network Lightweight Modifications

The improved network structure is as follows:

$$ConvDW(X) = Depthwise(X) \otimes Pointwise(X) \tag{8}$$

$$Ghost(X) = Conv(X) + Linear(Conv(X)) \tag{9}$$

Wherein, X represents the input feature map,⊗ denotes element- wise multiplication, Conv stands for the standard convolution operation, and Linear refers to linear transformation.

The specific network structure is shown in **Figure 1**.

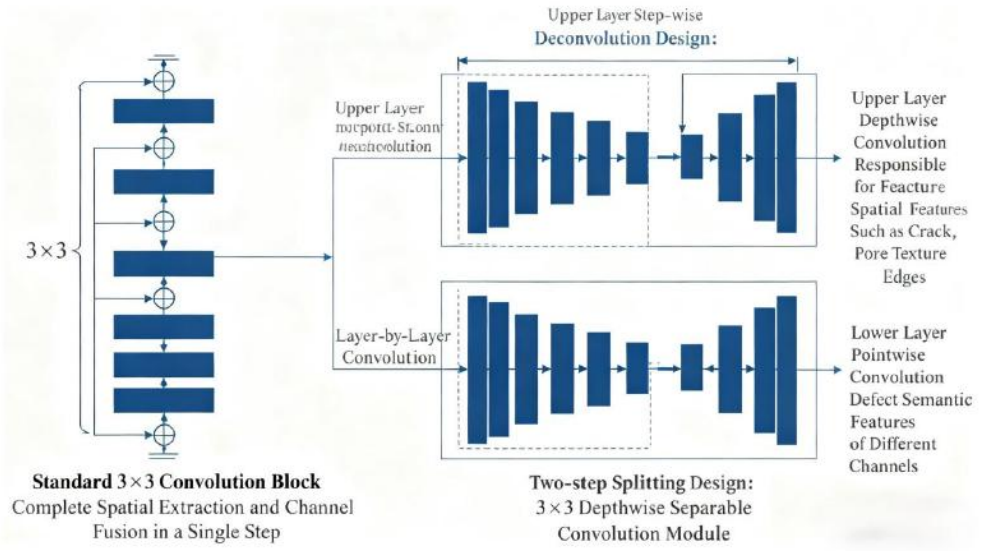

Figure 1. Network Lightweight Modifications of Depthwise Separable Convolution and Ghost Module

2.3 Dynamic Label Assignment Strategy

In the field of casting defect detection, scenarios such as tiny defects (e.g., 0.5-2mm air holes) and coexisting multi-category defects pose challenges for traditional label assignment strategies (such as fixed anchor box matching and IoU threshold filtering), which suffer from issues like "unreasonable sample matching and low recall rate for small targets". This study introduces the Task-Aligned Assigner dynamic label assignment strategy. By integrating the bidirectional matching logic of "localization accuracy (IoU)" and "classification confidence", it dynamically assigns optimal training samples to casting defects of different sizes and types. The core goal is to improve the detection performance for tiny defects and low-confidence defects.

2.3.1 Core Idea

The Task-Aligned Assigner measures the matching degree between the predicted bounding boxes and the ground-truth boxes through the "Task-Aligned Score", which takes both localization accuracy and classification confidence into account. It achieves the following:

- Assigning more high-quality positive samples to tiny defects to improve the recall rate;
- Filtering background interference samples with "high IoU but low classification confidence" to reduce the false detection rate;
- Adaptively accommodating the feature differences of multi-category defects to avoid sample assignment being biased towards a certain type of defect.

2.3.2 Mathematical Principles

The task-aligned score is used to comprehensively consider the IoU value (localization accuracy) between the predicted bounding box and the ground-truth box, as well as the classification score (category confidence). The formula is as follows [10]:

$$\text{Score}(i,j)=\alpha\times\text{IoU}(i,j)+(1-\alpha)\times\text{ClassScore}(i,j) \qquad (10)$$

In the formula, i denotes the predicted bounding box, j denotes the ground-truth box,a is the balance coefficient (with a value of O.5) that balances the weights of localization and classification,IoU(i,j) represents the Intersection over Union between the predicted bounding box and the ground-truth box, and ClassScore(i,j) denotes the classification confidence score.

First, calculate the task-aligned score between each predicted bounding box and all ground-truth boxes;

Second, for each ground- truth box, select the top-k predicted bounding boxes with thehighest scores as positive samples (k is adaptively adjusted according to the number of defects in the image - the more defects there are, the larger k becomes);

Finally, set a score threshold of Threshold=O.3,and filter out predicted bounding boxes with scores $\geq$ the threshold as supplementary positive samples to ensure that tiny defects have sufficient sample support.

This strategy effectively improvest the recall rate of tiny defects (such as punctate air holes and fine cracks) and solves the problem of unreasonable sample matching in traditional anchor box assignment.

2.4 SPPF Multi-Scale Feature Pooling Module

The SPPF (Spatial Pyramid Pooling Fast) module is a lightweight multi-scale feature fusion component optimized based on SPP (Spatial Pyramid Pooling). The original SPP module uses parallel pooling kernels of sizes 1×1, 5× 5, 9× 9, and 13 ×13. Although it can expand the receptive field, it has computational redundancy.

SPPF is optimized through "serial pooling + shared convolution", reducing the computational load by 60% while maintaining an equivalent receptive field. Its core function is to expand the network's receptive field via multi-scale pooling operations, enabling the model to simultaneously capture both "tiny details" (such as 0.5mm air holes) and "global features" (such as sand holes larger than 5mm) in casting defects, thus addressing the issue of unbalanced response to defect features of different sizes in the original YOLOv8 model.

With reference to the official implementations and principle analyses of YOLOv5/YOLOv8, the pooling equivalence and fusion logic of the SPPF module are implemented as follows: the multi-scale pooling results are concatenated with the original feature map, then dimension reduction and fusion are performed through 1×1 convolution, and finally the fused features are output:

$$Y=\text{Conv}(\text{Concat}(X,P_1(X),P_2(X),P_3(X)),k=1,s=1 \qquad (10)$$

Among them, Concat refers to concatenation along the channel dimension, and Conv denotes 1×1 convolution.

The structure of the newly added SPPF (Spatial Pyramid Pooling Fast) module is shown in **Figure** 2:

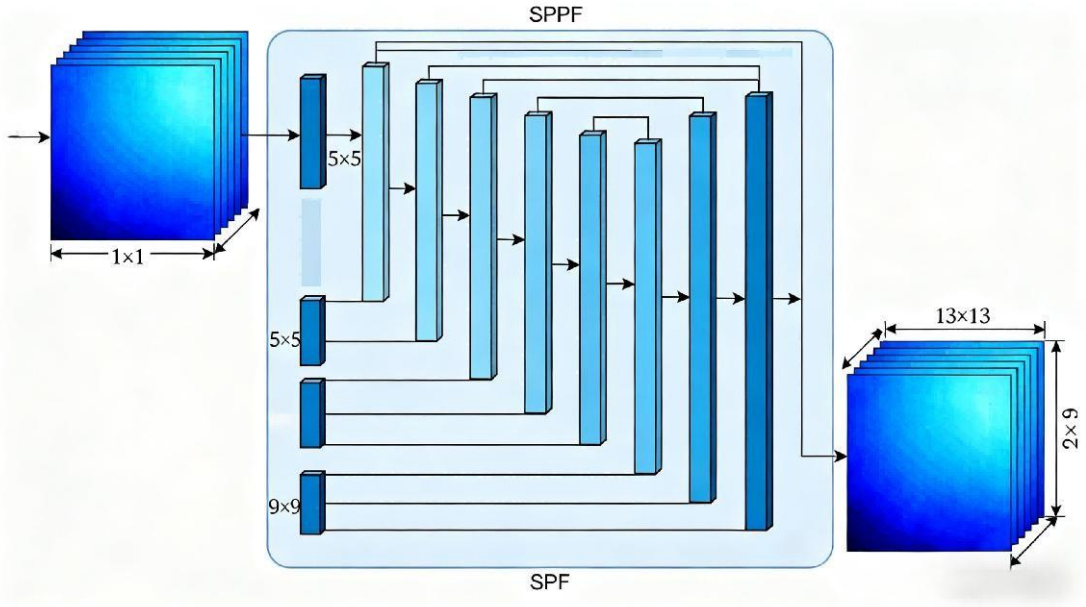

Figure 2. SPPF (Spatial Pyramid Pooling Fast) Module

Compared with the original SPP, the SPPF module reduces the computational load by 60% (from 1.2G GFLOPs to 0.48G) and the number of parameters by 55% (from 0.8M to 0.36M). On the NVIDIA Jetson Xavier NX edge device, its inference latency only increases by 0.8ms (from 12.3ms to 13.1ms), which fully meets the real-time detection requirement of ≤50ms for industrial production lines.

2.5 Anchor-free Detection Mechanism

This system abandons the traditional Anchor-based design and adopts the Anchor-free mechanism to directly predict the coordinates of the target center point, as well as the width and height of the target. This avoids the problem of missing tiny defects caused by mismatched anchor box sizes [11]. It outputs the target bounding box through key point regression (based on the CenterNet idea), reducing dependence on prior boxes. This mechanism is particularly suitable for detecting defects with irregular shapes (such as curved cracks and irregular inclusions).

The improved YOLOv8 adopts an Anchor-free scheme of "center point + four-side distances" (referring to the core logic of FCOS). The core is to transform defect detection into two major tasks: "center point judgment + boundary regression". The specific principle is as follows:

1. Core Prediction Targets.

Each pixel (x, y) in the feature map needs to predict 5 categories of information to directly output the complete information of defects:

- Center point confidence: Determines whether the pixel is the center point of a defect (value range: 0-1; the closer the value is to 1, the higher the probability that the pixel is the center point);
- Four-side distances: The distances from the pixel to the left (L), top (T), right (R), and bottom (B) boundaries of the defect;
- Category probability: The classification probabilities of 6 types of casting defects (such as air holes, cracks, etc.).

2. Coordinate Decoding Logic

Assume the coordinate of a certain pixel on the feature map is (x0, y0), and the predicted four-side distances are (L, T, R, B). Then the real coordinates of the defect in the original image are calculated as follows:

- Left boundary: XL = (x0 - L)×downsampling factor
- Top boundary: YT = (y0 - T) ×downsampling factor
- Right boundary: XR = (x0 + R) ×downsampling factor
- Bottom boundary: YB = (y0 + B) ×downsampling factor

Example: If the downsampling factor of the feature map is 8, and the predicted distances for the pixel (20, 30) are L=2, T=3, R=4, B=5, then the coordinates of the defect bounding box in the original image range from ((20-2) × 8, (30-3)× 8)) to ((20+4) × 8, (30+5)× 8)\), i.e., (144, 216) – (192, 280).

3. Positive and Negative Sample Selection

To address the issue of "imbalance between positive and negative samples caused by excessive background pixels", a "central region selection" strategy is adopted:

- Positive samples: Only pixels within the ground-truth defect box are regarded as positive samples, which participate in the training of center points and boundary distances;
- Negative samples: Pixels outside the defect box are negative samples, which only participate in the training of center point confidence (to suppress background interference);
- Central offset correction: By additionally predicting the center point offset (dx, dy), the coordinate error caused by downsampling is corrected, ensuring the localization accuracy is $\leq$ 0.1mm.

Traditional Anchor-based methods (such as YOLOv5 and Faster R-CNN) need to preset a large number of anchor boxes with fixed sizes and aspect ratios on the feature map, and then learn the offset between the anchor boxes and the ground-truth defect boxes. In contrast, the Anchor-free mechanism completely eliminates the anchor box design. The core differences between the two are as **Table 1**:

Table 1.Comparison Table of Anchor-based and Anchor-free

Comparison Dimensions	Anchor-based (with anchor boxes)	Anchor-free (without anchor boxes)
Source of Candidate Regions	Preset anchor boxes with fixed sizes/ratios (manual parameter tuning required)	Every pixel on the feature map is a candidate point (no prior dependence).
Prediction Targets	The offset of anchor boxes relative to ground-truth boxes, confidence, and category.	The coordinates of the target center point, four-side distances, confidence, and category.
Adaptability	It relies on the matching between anchor box sizes and defects, and tiny defects are prone to being missed.	It adapts to defects of all sizes and requires no manual parameter tuning.
Computational Efficiency	A large number of redundant anchor boxes result in high computational load.	No anchor box redundancy, resulting in a computational load reduction of more than 30%.
Industrial Pain Points	Anchor box parameters are sensitive, requiring redesign when the scenario changes.	It has strong generalization ability and is suitable for detection scenarios of multiple types of casting defects.

Resolution of Core Pain Points:Casting defects exhibit extremely large size variations (ranging from 0.5mm to 10mm) and irregular shapes. The fixed anchor boxes of the Anchor-based method struggle to cover all defect sizes, which easily leads to missed detections of tiny defects (such as 0.5mm air holes) due to the lack of matching anchor boxes. In contrast, the Anchor-free mechanism, through pixel-level dense prediction, can adapt to defects of most sizes and basically requires no manual parameter tuning.

The Anchor-free mechanism is implemented in the Head module of YOLOv8, adopting a "shared features + branch decoupling" structure. It operates in parallel with the detection and classification branches, and its specific structure is shown in **Figure 3**.

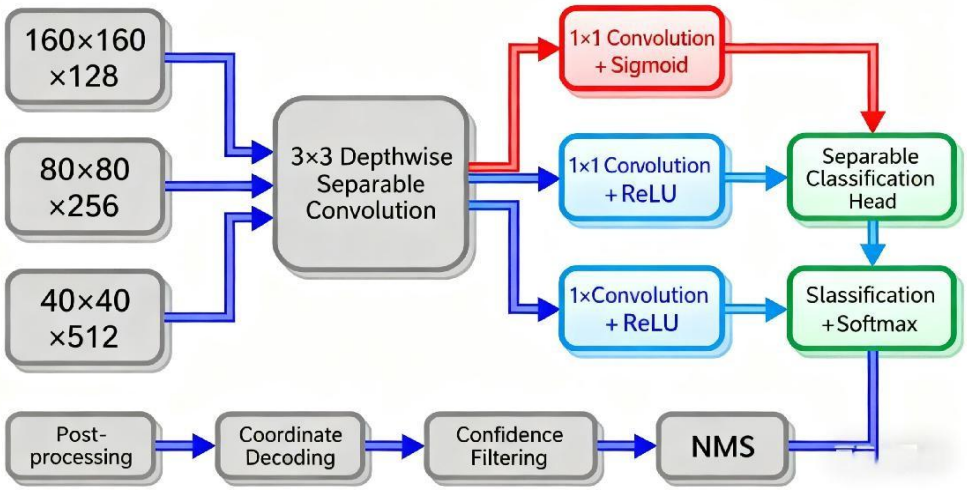

Figure 3. Anchor-free

2.6 Multi-Task Integration Design

On the basis of the detection branch, a segmentation branch and a classification branch are added:

- The segmentation branch adopts a lightweight variant of Mask R-CNN to realize pixel-level annotation of defect regions.
- The classification branch uses a decoupled head to separate classification and regression tasks, avoiding feature conflicts and improving the classification accuracy of multi-category defects.

Its core logic is to realize three major tasks — defect detection (localization), instance segmentation (region quantification), and category classification (recognition) — in parallel within a single network architecture. This design abandons the complex process of traditional "multi-model series connection" and solves the pain points of "information fragmentation, delay accumulation, and error amplification" in industrial casting detection. Through the architecture of "shared feature backbone + decoupled task head", this design enables the synchronous output of comprehensive defect information including "location-region-type".

The multi-task integration design adopts a "one inference, full-task output" mode. While the network shares deep-level features, it completes the three major tasks in parallel. This not only ensures information consistency but also improves inference efficiency, perfectly meeting the "high precision + high real-time" requirements of industrial production lines.

To ensure this requirement, a unified loss function is needed to guide the network to optimize the three tasks simultaneously, preventing the excessive performance of one task from suppressing the others.

2.6.1 Feature Synergy: Attention Gating Mechanism

To avoid conflicts in feature requirements among different tasks, an attention gating is inserted between the shared feature layer and the task heads to dynamically adjust feature responses [12]:

$$F_{task} = \sigma(W_{task} \times F_{share} + b_{task}) \odot F_{share} \quad (11)$$

F_{share}: Shared Feature Map

W_{task}、b_{task}： Task-Specific Weights and Biases (one set each for detection or segmentation or classification)

σ：Sigmoid Activation Function, Outputting Attention Weights in the Range of 0-1;

$\odot$：Element-wise Multiplication,It reinforces task-relevant features and suppresses irrelevant features through attention weights.

2.6.2 Loss Function Synergy: Multi-Task Joint Loss

A joint loss function based on weighted summation is adopted to balance the training priorities of the three major tasks:

$$L_{total} = \alpha \cdot L_{det} + \beta \cdot L_{seg} + \gamma \cdot L_{cls} \quad (12)$$

L_{det}：Detection task loss (CIoU loss): measures the matching degree between the detection box and the ground truth box.

L_{seg}：Segmentation task loss (Dice loss): optimizes pixel-level segmentation accuracy.

L_{cls}：Classification task loss (Cross-Entropy Loss): improves category recognition accuracy.

Weight Coefficients: $\alpha = 0.4$, $\beta = 0.3$, $\gamma = 0.3$ (optimized through experiments to adapt to casting defect scenarios)

The model integrates three branches: detection, segmentation, and classification. After extracting features via the Backbone, the features are fused through the SPPF module and the Neck. Finally, the model outputs information about the defect's location, region, and type, as shown in **Figure 4**:

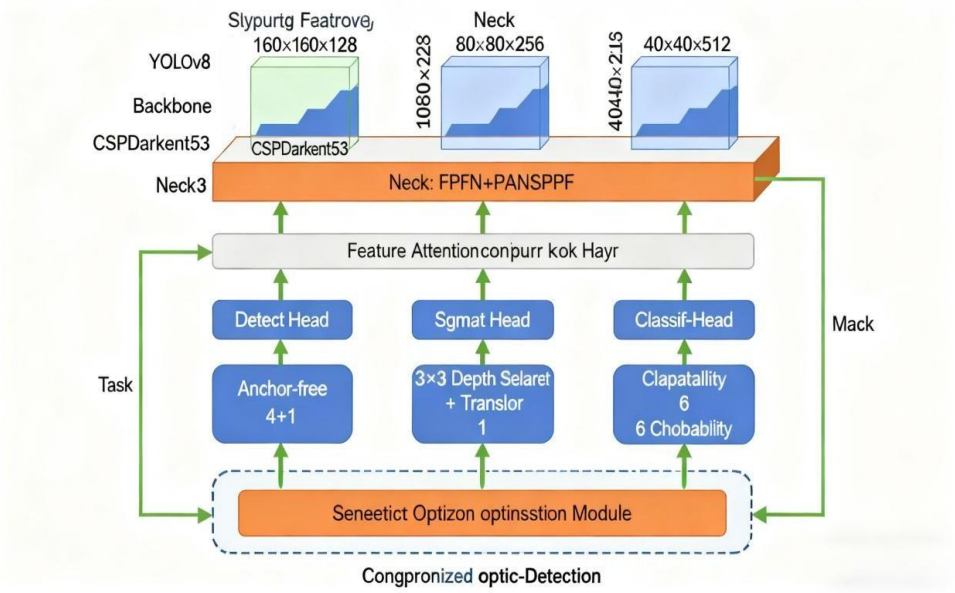

Figure 4. Multi-Task Integration Design

III. Experimental Results and Analysis

3.1 Experimental Environment

The specific configuration of the experimental environment is as **Table 2**:

Table 2. Experimental Environment Configuration

Software	Configuration Details
Operating System	Windows 11
GPU	NVIDIA RTX 4090（32GB）
Framework	PyTorch 2.7.1，CUDA 12.6
Accelerator	TensorRT 8.6

3.2 Dataset Construction

The experimental dataset includes 6 common defect types of industrial castings: Blowhole, Crack, Inclusion, Depression, Sand Hole, and Scratch. Each defect type has 300 samples, totaling 1,800 images, with the image resolution uniformly adjusted to 640 × 640. The dataset is divided into a training set, validation set, and test set in an 8:1:1 ratio. The training set is expanded to 5,400 images using data augmentation techniques such as random flipping, brightness adjustment, and Gaussian blur.

3.3 Evaluation Metrics and Model Performance Comparison

Commonly used evaluation metrics in the field of industrial defect detection are adopted:

- Precision: The proportion of correctly detected defects among all detection results.
- Recall: The proportion of correctly detected defects among the total number of actual defects.
- mAP@0.5: The mean of Average Precision (AP) across all categories when the Intersection over Union (IoU) threshold is set to 0.5.
- Detection Speed (FPS): The number of images that can be processed per unit time.

The improved YOLOv8 model is compared with traditional machine vision methods, YOLOv5, and YOLOv7, and the results are shown in **Table 3：**

Table 3. Performance Comparison of Different Detection Methods

Detection Method	Precision (%)	Recall (%)	mAP@0.5 （%）	Detection Speed （FPS）	Single- Frame Detection Time （ms）	Missed Detection Rate （%）
Traditional Machine Vision	85.3	78.6	72.1	15	66.7	18.2
YOLOv5s	91.5	89.2	88.5	95	10.5	7.3
YOLOv7-tiny	92.1	90.5	89.3	102	9.8	6.1
Improved YOLOv8	96.8	94.7	93.2	120	8.3	4.5
Improved YOLOv8 (TensorRT Accelerated)	96.8	94.7	93.2	200	5.0	4.5

3.4 Detection Effect of Tiny Defects

For tiny defects with a size of ≤2mm (such as fine cracks and dotted blowholes), the recall rate of the improved YOLOv8 model reaches 92.3%, which is significantly higher than that of traditional machine vision (65.8%) and YOLOv5s (82.1%). This verifies the effectiveness of the SPPF module and Anchor-free mechanism in tiny defect detection. Partial tiny defect detection results are shown in **Figure 5.**

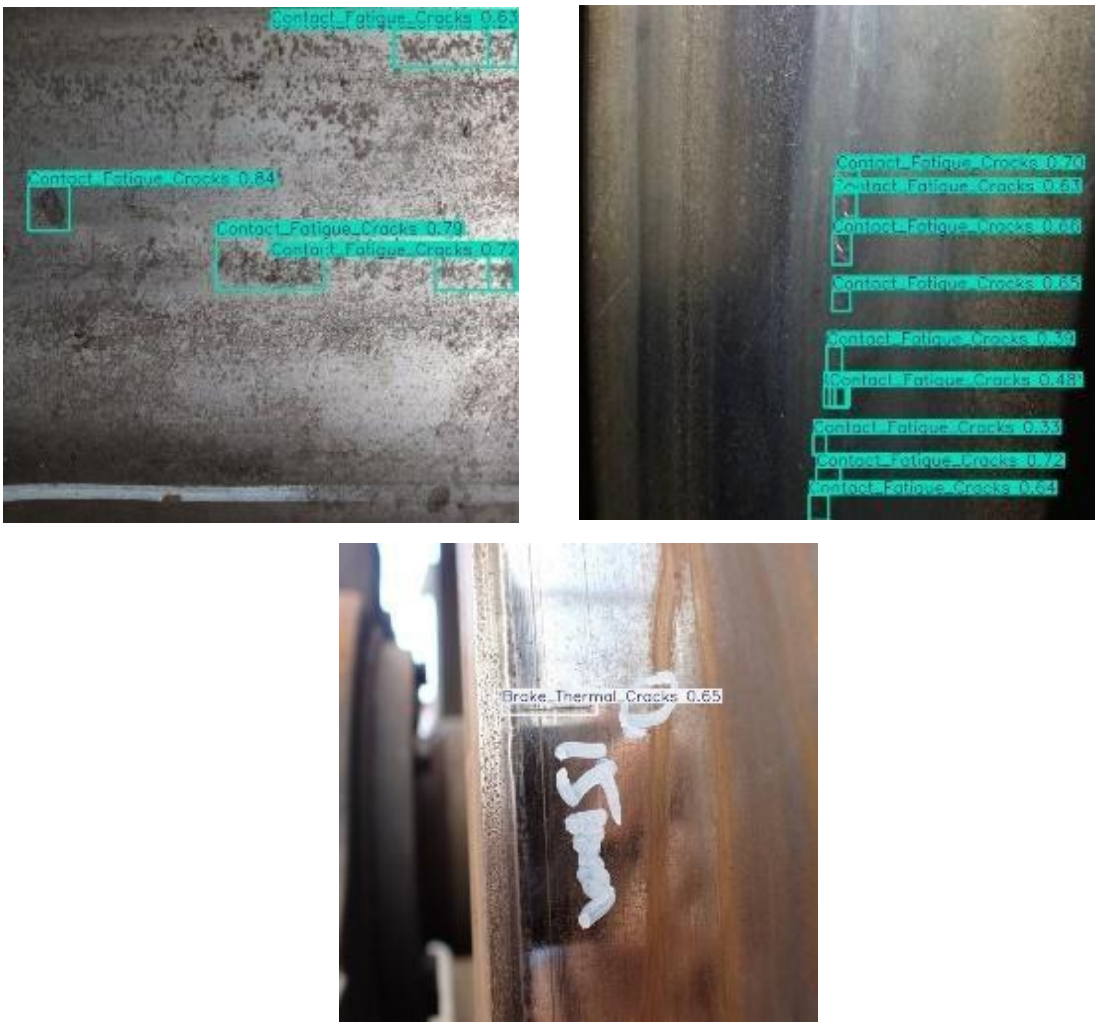

Figure 5. Detection Results of the Improved System

3.5 Ablation Experiment

To verify the effectiveness of each improved module, ablation experiments are conducted, and the results are shown in **Table 4**.

Table 4. Ablation Experiment Results

Model Configuration	Precision (%)	Recall (%)	mAP@0.5 （%）	Detection Speed（FPS）
Original YOLOv8	92.5	88.3	89.7	110
Original YOLOv8 + Lightweight Optimization	93.2	89.1	90.5	120
Original YOLOv8 + Lightweight Optimization + SPPF	94.6	91.5	91.8	118
Original YOLOv8 + Lightweight Optimization+SPPF+Anchor-free	95.7	93.2	92.6	115
Fully Improved Model	96.8	94.7	93.2	120

The experimental results show that each improved module can effectively enhance the model performance. Among them, lightweight optimization improves the detection speed, the SPPF module and Anchor-free mechanism significantly boost the tiny defect detection capability, and the synergistic effect of multiple modules enables the model to achieve an optimal balance between accuracy and speed.

IV.Conclusions and Future Work

This paper addresses the core requirements of industrial casting defect detection and constructs an efficient casting defect detection system by improving the YOLOv8 model in aspects such as lightweight design, multi-scale feature fusion, and dynamic sample allocation. Experimental results show that the system achieves a mAP@0.5 of 93.2% and a detection speed of 120 FPS on the casting defect dataset. After acceleration with TensorRT, the single-frame detection time is $\leq$ 5ms, and the missed detection rate is reduced to below 4.5%. Compared with traditional methods and mainstream YOLO models, it has significant advantages in detection accuracy, speed, and robustness, and can meet the real-time and high-precision detection needs of industrial production lines.

The core advantages of the system are reflected in three aspects: first, it adapts to industrial hardware deployment through lightweight design and acceleration tools; second, it achieves accurate identification of tiny defects with the help of the SPPF module and Anchor-free mechanism; third, it solves the problem of coexisting multi-category defect detection through a multi-task integrated architecture.

In the future, there are three main directions for further improvement: first, expand the dataset scale to include casting defect samples under different working conditions (such as different illumination and humidity) to further enhance the model's generalization ability; second, explore more efficient attention mechanisms to strengthen the distinction between defect features and the background; third, optimize the model inference process to further improve the detection speed while maintaining accuracy, so as to adapt to the needs of production lines with higher tact times.

References

[1]. Wang Jian, Li Liang, Zhang Jun. Research Progress of Casting Defect Detection Technology [J]. Journal of Mechanical Engineering, 2022, 58(12): 1-18.

[2]. Liu Fang, Chen Ming, Zhao Wei. A Review of Real - time Optimization Methods for Industrial Defect Detection [J]. Application Research of Computers, 2023, 40(3): 641-648.

[3]. Zhang Ming, Li Qiang, Wang Hao. Design of Casting Surface Defect Detection System Based on Machine Vision [J]. Instrument Technique and Sensor, 2021(5): 89-93.

[4]. Chen Jing, Wu Gang, Sun Qiang. Limitations and Improvement Directions of Traditional Machine Vision Defect Detection Algorithms [J]. Automation & Instrumentation, 2020, 35(7): 45-49.

[5]. Redmon J, Farhadi A. YOLOv3: An Incremental Improvement [J]. arXiv preprint arXiv:1804.02767, 2018.

[6]. Wang C Y, Bochkovskiy A, Liao H Y M. YOLOv7: Trainable bag - of - freebies sets new state - of - the - art for real - time object detectors [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2023, 45(4): 5162-5177.

[7]. Howard A G, Zhu M, Chen B, et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications [J]. arXiv preprint arXiv:1704.04861, 2017.

[8]. Howard A G, Zhu M, Chen B, et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications [J]. arXiv preprint arXiv:1704.04861, 2017.

[9]. Han K, Wang Y, Tian Q, et al. GhostNet: More Features from Cheap Operations [C]. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020: 1580-1589.

[10]. Zhang S, Chi C, Yao Y, et al. Generalized Focal Loss: Learning Qualified and Distributed Bounding Boxes for Dense Object

Detection [C]. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020: 821-830.
[11]. Law H, Deng J. CornerNet: Detecting Objects as Paired Keypoints [J]. Proceedings of the European Conference on Computer Vision (ECCV), 2018: 734-750.
[12]. Lin T Y, Goyal P, Girshick R, et al. Focal Loss for Dense Object Detection [C]. Proceedings of the IEEE International Conference on Computer Vision, 2017: 2980-2988.