Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

# MTF-YOLO: A Steel Surface Defect Detector with Multiscale Texture Fusion and Variance-Guided Convolution

**Delin Huang**
Shanghai Second Polytechnic University
**Guo Gong**
Shanghai Second Polytechnic University
**Jinghui Yang**
jhyang@sspu.edu.cn
Shanghai Second Polytechnic University
**Dexian Wang**
Shanghai Second Polytechnic University
**Linlin Liao**
Shanghai Second Polytechnic University

**Research Article**

**Keywords:** steel surface, surface defect detection, YOLO, multi-scale and multi-level feature fusion

**Posted Date:** June 11th, 2026

**DOI:** https://doi.org/10.21203/rs.3.rs-8234620/v1

**License:** This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

**Additional Declarations:** No competing interests reported.

# MTF-YOLO: A Steel Surface Defect Detector with Multiscale Texture Fusion and Variance-Guided Convolution

Delin Huang [a], Guo Gong [a], Jinghui Yang [a,b,*], Dexian Wang [a], Linlin Liao [a]

[a] School of Intelligent Manufacturing and Control Engineering, Shanghai Polytechnic University, Shanghai 201209, Chinaa
[b] Wuyi Intelligent Manufacturing Industrial Technology Research Institute, Zhejiang 321200, China

Abstract

In industrial surface inspection, defects are often small, weakly textured, and bounded by uncertain edges—conditions under which state-of-the-art detectors struggle, especially in low-contrast scenes and for thin cracks. To address these challenges, this paper proposes MTF-YOLO, a YOLO-based detector that integrates three modules: Defect Variance-Guided Convolution (DVGC), Defect Edge Enhancement (DEE), and Defect-Oriented Texture Fusion (DOTF). Firstly, DVGC explicitly models the statistical disparity between foreground and background, improving separability for small, low-contrast targets. Secondly, DEE sharpens contours and fine details at the feature-level, mitigating localization drift caused by blurred boundaries. Thirdly, DOTF aggregates texture cues across multiple scales and orientations. In addition, this paper employs a Path Aggregation Network–FPN (PAN–FPN) neck together with Spatial Pyramid Pooling–Fast (SPPF) to efficiently fuse semantic and low-level information across pyramid levels. Experiments show that MTF-YOLO achieves mAP@0.5 scores of 84.1% and 82.7% on NEU-DET and GC10-DET, respectively; it strikes a favorable balance among accuracy, efficiency, and deployability, and demonstrates robust performance for steel surface defect detection.

Keywords: steel surface; surface defect detection; YOLO; multi-scale and multi-level feature fusion

## 1. Introduction

Steel, noted for its high specific strength, excellent formability and machinability, and controllable cost, has long been foundational to the shipbuilding, aerospace, and heavy equipment industries. Its surface quality directly determines component reliability and service life [1]. As smart manufacturing scales from pilot deployments to full production, production lines increasingly demand consistent, stable, and high-precision online inspection; consequently, automated surface defect detection has become a critical component of quality assurance [2]. Among steel surface defects, small-object categories typically appear as fragmented, low-contrast, multi-directional, and multi-scale textures that blend easily with background patterns, increasing both false-positive and false-negative rates [3]. Manual visual inspection depends heavily on expert experience and sustained attention, making it difficult to ensure efficiency, reproducibility, and coverage under high-speed, long-duration online monitoring [4].

Computer vision provides a practical engineering route. Convolutional neural network (CNN)-based detectors offer robust feature representations in industrial scenarios [5], and a representative one-stage detector, YOLOv8, provides an attractive accuracy–speed–parameter trade-off in both production deployments and public benchmarks [6]. Nevertheless, cross-scale alignment and detail-sensitive representations remain primary bottlenecks when confronting defects with fine textures and a wide range of scales [7]. Vision Transformers (ViTs) excel at modeling long-range dependencies [8], and the Detection Transformer (DETR) removes non-maximum suppression (NMS) while enabling end-to-end inference [9]; however, in defect-detection settings with a high ratio of small objects and sparse foregrounds, convergence efficiency and the computational overhead of global attention need careful balancing [10].

To tackle these challenges, this paper analyzes the characteristics and detection difficulties of small-object defects and presents targeted optimizations based on YOLOv8:

(1) Defect Variance-Guided Convolution (DVGC): employs learnable local statistics to adaptively modulate convolutional responses, emphasizing defect-relevant regions while suppressing background noise; a selective feature-extraction path reduces computation.

(2) Defect Edge Enhancement (DEE): introduces a lightweight, edge-aware enhancement that leverages multi-scale edge operators and gated fusion to preserve continuous boundaries for subsequent multi-scale aggregation.

(3) Defect-Oriented Texture Fusion (DOTF): integrates multi-directional filtering with attention-based fusion in the neck to capture orientation-dependent textures of small defects, while a gated residual formulation preserves fine details under perturbations.

Comprehensive experiments on multiple steel-surface datasets demonstrate consistent gains in small-object detection accuracy at comparable parameter counts and latency. The remainder of this paper is organized as follows: Section 2 reviews prior work on defect and object detection; Section 3 details the three core modules and their implementation; Section 4 reports experiments and analyses; Section 5 concludes and outlines future directions.

## 2. Related work

### 2.1 YOLOv8

Since the emergence of one-stage detectors, the YOLO family has evolved from anchor priors to point-based

∗Corresponding author: Jinghui Yang, email: jhyang@sspu.edu.cn

(anchor-free) candidate generation and from coupled to decoupled heads. In parallel, CSP-style C2f backbones and PAN–FPN necks enable multi-scale processing, color augmentation, and random patching strategies, yielding three-scale outputs with different receptive fields [11]. YOLOv8 adopts an anchor-free design with a decoupled head, IoU-based and distributional regression losses, and dynamic label assignment, which improves convergence and boundary quality [12].

### 2.2 Statistics-Guided Convolution for Defect Detection

Local statistics are effective for highlighting textures and edges in cluttered backgrounds. Early studies relied on gradient- and Laplacian-based priors with fixed edge emphasis [13]. Subsequent work introduced sliding-window mean and variance to modulate responses—treating high variance as structure and low variance as flat noise—while channel reweighting and spatial attention offered lightweight saliency that coordinates priors and semantics [14]. In detection, these statistics are commonly integrated with multi-scale fusion to stabilize boundaries and textures [15]. Recent approaches replace hard thresholds with learnable statistical mappings that directly modulate convolutional responses, balancing noise suppression and detail retention [16].

### 2.3 Edge-Aware Multi-Scale Fusion for Defect Detection

Defect edge enhancement often follows a "contour prior + multi-scale fusion" paradigm. Classical Sobel- and Laplacian-type filters highlight boundaries in low-contrast environments [17]. Later approaches implement boundary-aware branches via lightweight or deformable convolutions to propagate details across scales while avoiding discontinuities [18]. Decoupled heads with IoU-oriented loss functions further stabilize edge alignment [19]. PAN–FPN paths that incorporate dilated and depthwise-separable kernels expand the receptive field and improve detection performance [20]. More recent work employs directional attention to gate contour responses and suppress noise in cluttered backgrounds [21].

### 2.4 Direction-Aware Texture Fusion for Defect Detection

Texture-dominant defects are typically addressed with direction-sensitive modeling and inter-layer fusion. Gabor-type filters capture oriented patterns such as stripes and grooves, but show limited robustness to large-scale and rotation changes [22]. Structure-tensor formulations and gradient-histogram descriptors improve directional consistency, yet may break down under weak textures and noisy conditions [23]. Deep models aggregate texture features via PAN–FPN-style necks that incorporate directional attention mechanisms and anisotropic convolutions; learned weights suppress background activations and emphasize defect-specific textures [24]. Recent designs combine multi-directional filtering with attention-based fusion and residual propagation to preserve fine textures under perturbations [25].

## 3. Method

### 3.1 Overall Architecture Design

This study enhances the YOLOv8 framework to address the slender morphologies and intricate textures characteristic of industrial surface defects. The architecture processes 640 × 640 inputs through a backbone that reduces the spatial resolution to 20 × 20 at stride 32 while expanding the channel dimension to 1024. The neck adopts a PAN–FPN structure for semantic alignment, and the detection head operates on feature maps P3–P5. Three specialized modules are integrated to alleviate representation bottlenecks: DVGC, placed after the first C2f stage, strengthens crack connectivity through variance-guided modulation; DEE, inserted after the SPPF block, consolidates edge information using hexagonal kernels; and DOTF, located at the P3 level, fuses eight directional filters with depthwise convolutions to form a multi-oriented texture representation. This design balances detection sensitivity and noise suppression, as illustrated in Fig. 1.

### 3.2 Defect Variance-Guided Conv （DVGC）

Traditional single-path FPNs only relabel features through channels or in spatial dimensions, while fixed Sobel/Laplacian edges exhibit poor generalization. To address these issues, this module processes input X through two parallel paths: the primary input enters the main convolutional layer to generate convolutional outputs. Simultaneously, local variance is used as input for the variance-guided network, which generates guidance weights. These weights are then processed by CrazingBoost to produce enhancement coefficients. Feature fusion applies this coefficient to the convolutional output to form an integrated result. The dual residual connection path preserves low-frequency content while stabilizing the learning process. The final output emphasizes weak boundaries and low contrast microcracks while suppressing noise amplification effects across different scenes, as shown in Fig .2.

To enhance the separability of weak boundaries and fine cracks, we propose the Defect Variance-Guided Convolution

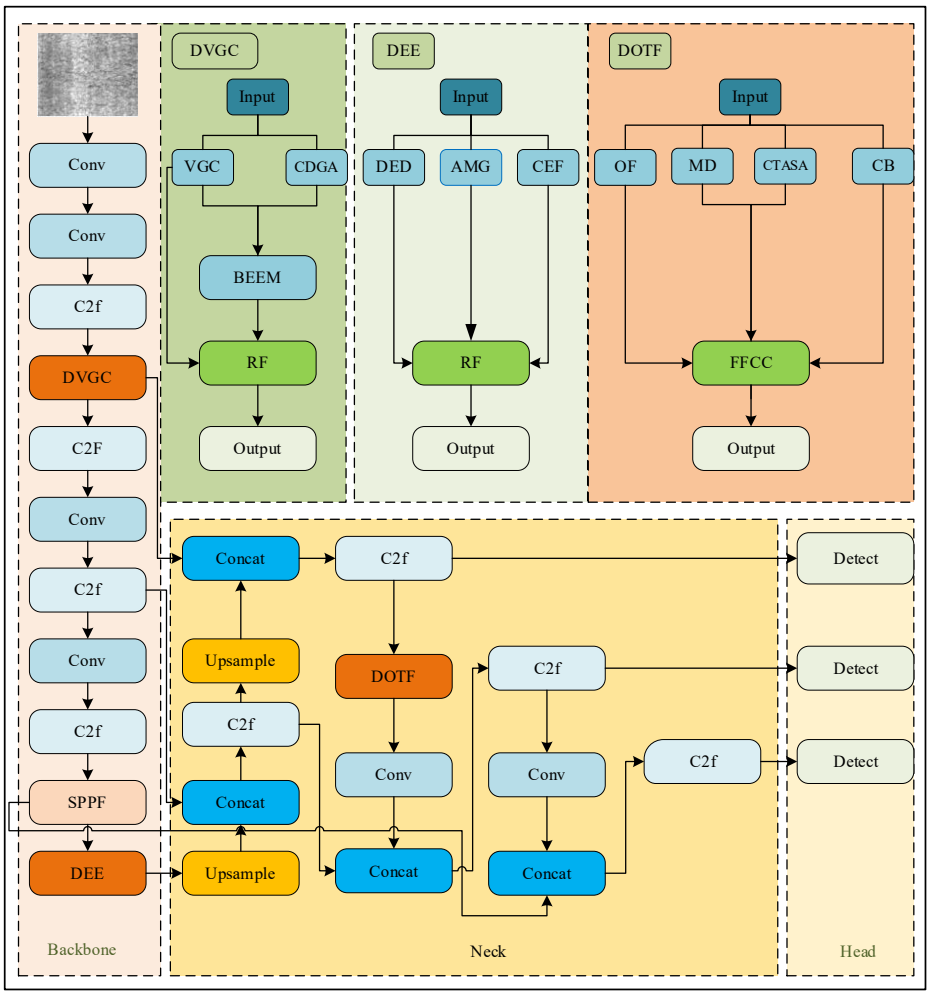



Fig. 1 MTF-YOLO Architectur

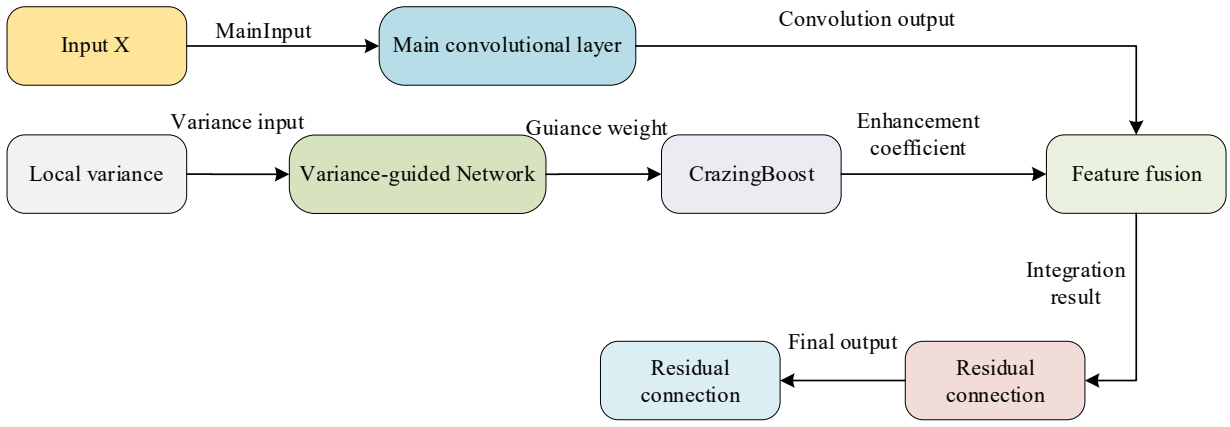



Fig. 2 Structure of the Defect Variance-Guided Convolution Module

Module (Defect-Variance-Guided-Conv). Given an input feature X, the module runs two parallel paths: a main convolution branch producing the morphological-textural baseline F(X), and a variance-guided branch that computes local variance within the neighborhood N(p):

$$D(p) = Var(X_{N(p)}) \tag{1}$$

D then undergoes normalization φ(·) followed by a lightweight gated mapping θ(·) (where θ is typically a sigmoid function) to generate positional weights, yielding:

$$W = \theta(\varphi(D)), W \in [0,1]^{\{H*W\}} \tag{2}$$

The final output employs position-adaptive amplification with a dual residual topology, where ⊙ denotes the element-wise product:

$$Y = X + F(X) + (W \mathrm{e}\ F(X)) \quad (3)$$

The short residual F(X) stabilizes gradients, while the long residual X preserves low-frequency context. This design enables adaptive enhancement of high-variance defect regions and suppresses background over-amplification without relying on fixed priors or edge operators (e.g., Sobel/Laplacian). Unlike channel attention mechanisms or FPN stacking approaches, the proposed module employs a learnable variance map D as guidance. Experimental results demonstrate that the module improves mAP50 and Recall in ablation studies. The processing sequence follows: local variance computation, guided weight generation, enhancement coefficient calculation, residual fusion. Specifically designed for metal surface defect detection, the module effectively improves the localization of fine cracks while maintaining controllable inference cost during training.

### 3.3 Defect Edge Enhancement Module (DEE)

Traditional detectors combine single-path convolutions, FPN stacking, or SE/CBAM attention mechanisms, sometimes supplemented by Sobel/Laplacian mapping. However, weak boundaries still disappear after downsampling. To address these limitations, this module employs a dual-path architecture with input X.

The upper path generates edge features through six-directional edge detection (horizontal, vertical, and four diagonal directions), followed by edge-list construction, edge stacking, and maximum fusion to produce an edge map E_{edge}.

The attention path processes attention input through an attention mechanism and a main projection layer to yield projection features F(X) and enhancement coefficients. The enhancement coefficients are further refined through a crazing enhancement step to produce refined enhancement coefficients.

Subsequently, a feature fusion module merges the projection features F(X), the refined enhancement coefficients from the crazing enhancement step, and the edge map E_{edge} from the upper path. Finally, a residual connection combines the fused features with the original input to produce the final output. The module architecture is illustrated in Fig. 3.

The attention mechanism computes a positional variance map D(p) over neighborhood N(p) for characterizing local feature variability:

$$D(p) = Var\{X_\{N(p\}\} \quad (4)$$

D then undergoes normalization followed by a lightweight gated mapping φ(·) to generate positional weights:

$$w = \sigma(\phi(D)), w \in [0,1]^{\{H \times W \times 1\}} \quad (5)$$

where σ(·) denotes the sigmoid function and φ(·) is a lightweight convolutional mapping (typically a 1×1 convolution followed by batch normalization). To control the reinforcement strength, an adjustable coefficient γ > 0 is introduced to construct the reinforcement coefficient:

$$w = \sigma(\phi(D)), w \in [0,1]^{\{H \times W \times 1\}} \quad (6)$$

The enhancement coefficients are refined through a crazing enhancement step. Specifically, the edge map E_{edge} from the edge detection branch is used to compute a crazing enhancement coefficient C:

$$C = 1 + \sigma(E_\{edge\} \cdot mean) \quad (7)$$

where σ(·) denotes the sigmoid function and the mean operation is performed over the channel dimension.

The feature-fusion path integrates the projection features F(X) from the main projection layer with the attention weights A from the attention mechanism, the crazing enhancement coefficient C, and a learnable boost parameter B. A residual scaling factor α is introduced to control the base amplification level. The final output employs position-adaptive amplification with a dual residual topology:

$$Y = F(X) \cdot [\alpha + A \cdot C \cdot B] \quad (8)$$

Through the synergy of "six-directional edge priors + attention projection gain," this module directionally amplifies low-contrast regions like fine cracks. Directional maximum fusion, constrained by a scaled sigmoid function, suppresses noisy enhancement coefficients, ensuring stability. A terminal residual provides a fallback path, balancing interpretability and robustness. Computational overhead is minimal, with a low parameter count allowing easy integration into existing backbones.

### 3.4 Directional Texture Fusion Module (DOTF)

Traditional detectors employ single-branch, FPN stacking, or SE/CBAM recalibration mechanisms, lacking directional boundary and texture priors. Fine cracks disappear after subsampling, and attention mechanisms may amplify noise.

The proposed module integrates multi-source priors, attention-weighted fusion, and lightweight residual enhancement. Input feature maps are decomposed along three paths: (i) basic feature projection preserves low-frequency shape cues; (ii) directional texture filters, guided by learnable texture weights, extract horizontal, vertical, and diagonal textures; and (iii) scale feature extraction captures multi-scale details. A multi-feature fusion block aggregates the three streams to form an integrated feature map. The integrated feature map is then processed by SE attention mechanisms to generate attention weights, which are applied to produce enhanced features. Simultaneously, texture analyzers output learnable texture weights for the texture-enhancement unit. The enhanced features and texture weights are combined in the texture-enhancement unit to deliver the final result, as shown in Fig. 4

To fuse and enhance the multi-dimensional features, we design multi-branch fusion, SE attention re-calibration, texture priors with quadratic reinforcement, and residual outputs. Let the input features be X.

Multi-branch Fusion. Basic features are produced by 1×1 convolutions:

$$F_{base} = Conv_{1\times1}(X) \quad (9)$$

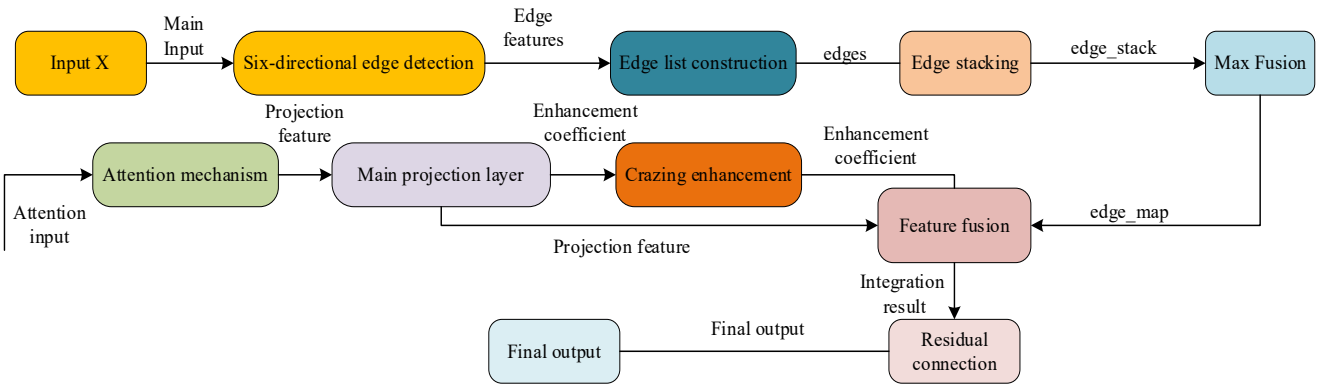



Fig. 3 Defect Edge Enhancement Module

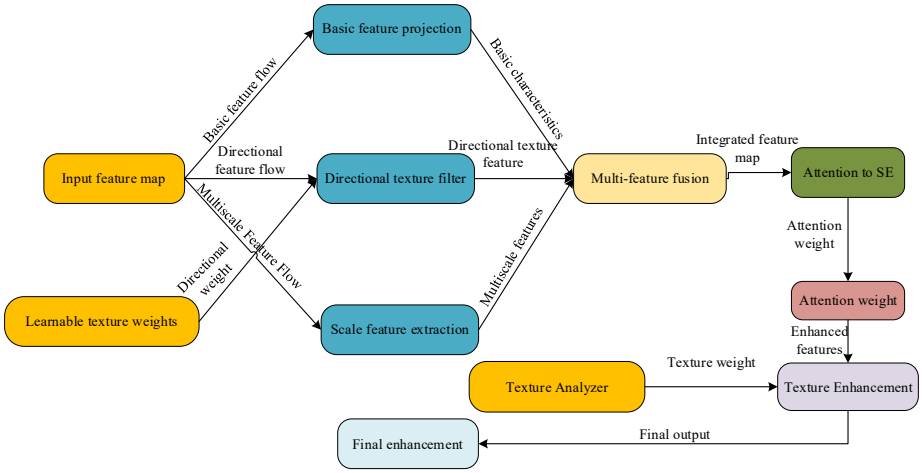



Fig. 4 Defect-Oriented Texture Fusion Module

Directional features are obtained by convolving X with kernels $k_d \in D$ ,taking absolute values, and summing:

$$F_{dir} = \sum_{k_d \in D} |K_d * X| \quad (10)$$

Scale features sum the responses of X to kernels $k_s \in S$ :

$$F_{scale} = \sum_{Ks \in S} X * K_s \quad (11)$$

Concatenating the three streams and applying a 1×1 convolution yields the integrated feature map:

$$F_{\text{int}} = \text{Conv}_{1\times1}([F_{\text{base}}, F_{\text{dir}}, F_{\text{scale}}]) \quad (12)$$

SE Attention Re-calibration. Channel-level attention is applied to $F_{\text{int}}$ ,Global average pooling gives channel statistics $Z_c = GAP(F_{int}, c)$ :

$$Z_c = \frac{1}{HW} \sum F_{int,c}(i,j) \quad (13)$$

Two linear layers with activation functions $\delta(\cdot)$ (e.g., ReLU) and $\sigma(\cdot)$ $(sigmoid)$ map Z to channel weights $a_c$ :

$$\alpha = \sigma(W_2 \delta(W_1 z)) \quad (14)$$

Finally, each channel of $F_{int}$ is rescaled as $F_{\text{int},c} = a_c \cdot F_{\text{int},c}$ ,Texture Prior and Secondary Enhancement. A texture analyzer $\psi(\cdot)$ (MLP or 1×1convolution) processes feature statistics $\text{Stats}(X)$ to obtain a texture prior $\tau = \psi(\text{Stats}(X))$. With an adjustable coefficient $\gamma > 0$ ,secondary enhancement multiplies $F_{\text{int}}$ by $1 + \gamma\tau$ :

$$F_{tex} = F_{int} \square (1 + \gamma\tau) \quad (15)$$

This module captures directional textures, models multi-scale features, recalibrates channels via SE attention, and stabilizes learning with residual outputs, enabling portable crack detection across platforms.

## 4. Experimental Verification

### 4.1 Experimental Setup

To ensure the validity of this model comparison, all methods were run on the same platform and under the same training workflow, as detailed in Table 1.

Table 1 Hardware/Software Configuration and TrainingStrategy

| | | |
|---|---|---|
| Hardware | CPU | IntelXeonGold 5418Y |
| | RAM | 256GB |
| | GPU | NVIDIA A10 |
| | VARM | 24GB |
| Software | Python | 18.04 |
| | Pytorch | 2.7.0 |
| TrainingStrategy | Batch sizae | 8 |
| | Epoch | 300 |
| | Learning rate | 0.0001 |
| | LearningrateStrategy | cosine |

### 4.2 Dataset and Evaluation Metrics

To evaluate the adaptability and robustness of this configuration in industrial surface defect scenarios, we selected the NEU-DET and GC10-DET datasets for unified experiments. Samples were divided into training, validation, and test sets in an 8:1:1 ratio based on the non-overlap

principle. Performance metrics focused on mAP@50 and mAP@50:95, while also reporting Recall and F1 scores. To evaluate deployability, the number of parameters and FLOPs were additionally quantified. All experiments were reproduced under consistent hardware and threshold settings to ensure fair comparisons.

#### 4.2.1 NEU-DET Dataset

This study utilizes the NEU-DET steel surface defect dataset, covering six target categories: cracks (Cr), inclusions (In), patches (Pa), pitting surfaces (PS), rolling scale (RS), and scratches (Sc). The data are balanced across categories (300 images per category, totaling 1800 images) and partitioned into training/validation/test sets at an 8:1:1 ratio (1440/180/180), as illustrated in Fig. 5.

#### 4.2.2 GC10-DET Dataset

The GC10-DET dataset originates from a surface defect collection in real industrial scenarios. In this study, we carefully selected seven of the most representative defect types from the original dataset for experimentation and analysis: crescent gouges (Cg), oil stains (Os), punches (Pu), thread marks (Ss), waist folds (Wf), water stains (Ws), and weld lines (Wl). The final GC10-DET dataset comprises 2,001 valid images, with 1,599 images allocated to the training set, 201 images designated for validation, and 201 images designated for testing.

To evaluate the proposed image standardization and annotation file matching, the dataset partitioning adheres to an approximately 8:1:1 ratio allocation strategy (7.96:1.00:1.00), ensuring scientific rigor and reliability in model training, validation, and testing, as illustrated in Fig. 6.

Without altering the data or training workflow, we uniformly report AP, mAP, parameters, and FLOPs (calculated at the default inference resolution).

Precision and Recall Definitions:

$$Precision = \frac{TP}{TP + FP} \tag{16}$$

$$Recall = \frac{TP}{TP + FN} \tag{17}$$

Single-class average precision and the multi-class mean:

$$AP = \int_0^1 p(r)dr \tag{18}$$

$$mAP = \frac{\sum AP}{N}, N = 6 \tag{19}$$

F1 Composite Index:

$$F1 = 2 \times \frac{Precision \cdot Recall}{Precision + Recall} \tag{20}$$

TP denotes the number of correctly detected targets; all detections correspond to the total number of predicted boxes; GTBox indicates the total number of ground-truth boxes; p(r) corresponds to the Precision–Recall curve.

### 4.3 Comparative Experiment

#### 4.3.1 NEU-DET Results

This study evaluates MTF-YOLO against diverse detectors. Baselines include Faster R-CNN and RetinaNet; YOLO-series variants (YOLOv8, YOLOv11, YOLOv12 with −n, −s, −m); Transformer models Fous-DETR and FC-DETR; and specialized Es-NET and CABF-FCOS (Table 2).MTF-YOLO achieves mAP@0.5 of 84.1%, mAP@0.75 of 42.51%, an F1-score of 76.28%, and recall of 80.1%, with 35M parameters and 89 GFLOPs, balancing accuracy and efficiency. Versus RetinaNet, mAP@0.5 rises by 5.18 points (78.92→84.1) while using fewer parameters (35M vs 64.43M) and FLOPs (89 vs 181.5). Versus Faster R-CNN, mAP@0.5 improves by 8.65 points (75.45→84.1). Within the YOLO family, MTF-YOLO leads mAP@0.5, surpassing YOLOv8-s (78.24) and YOLOv12-m (78.31) by 5.86 and 5.79. It also exceeds YOLOv8-m (36.08) and YOLOv12-s (42.18) on mAP@0.75.

Compared with Transformer-based detectors, it tops mAP@0.5 (84.1 vs FC-DETR 83.4; Fous-DETR 79.5) while using fewer resources. Against Es-NET (79.47) and CABF-FCOS (76.6), MTF-YOLO shows higher mAP@0.5 and lower complexity. as shown in Fig. 7.

#### 4.3.2 GC10-DET Results

This study benchmarks MTF-YOLO against diverse detectors: Faster R-CNN and RetinaNet; YOLO variants (YOLOv8, YOLOv11, YOLOv12 with −n/−s/−m configurations); and Transformer models Fous-DETR and FC-DETR, as detailed in Table 3. MTF-YOLO achieves mAP@0.5 of 82.86%, mAP@0.75 of 40.21%, an F1-score of 81.2%, and recall of 75.3%. Versus RetinaNet, mAP@0.5 increases by 4.38 points (78.48% → 82.86%); RetinaNet's mAP@0.75 is 33.4%, which is 6.86 points lower than ours. Versus Faster R-CNN, mAP@0.5 rises by 5.99 points (76.87% → 82.86%). Within the YOLO family, MTF-YOLO ranks first in mAP@0.5; it exceeds YOLOv8-m (74.27%) and YOLOv12-m (70.24%) by 8.59 and 12.62 points; on mAP@0.75, it surpasses YOLOv8-m (38.05%) and YOLOv12-m (34.38%) by 2.16 and 5.83 points. Compared with Transformer-based detectors, MTF-YOLO leads in mAP@0.5 (82.86% vs FC-DETR 81.63%, Fous-DETR 82.31%). Overall, MTF-YOLO demonstrates strong performance on mAP@0.5 and practical deployability, as shown in Fig. 8. Results indicate robust detection accuracy across metrics while maintaining balanced efficiency and reliability in practice

### 4.4 Ablation Experiment for NEU-DET

This section presents a comprehensive ablation study to verify the effectiveness of each module. The cases in Table 4 are labeled a–h and are shown in Fig. 9 with the same ordering. Case a is the baseline YOLOv8-n. Cases b–h are obtained by successively adding modules to the baseline: b (+DVGC), c (+DEE), d (+DOTF), e (+DVGC+DEE), f (+DVGC+DOTF), g (+DEE+DOTF), and h (+DVGC+DEE+DOTF). Each module brings a performance gain, and combinations provide further improvements over the baseline.

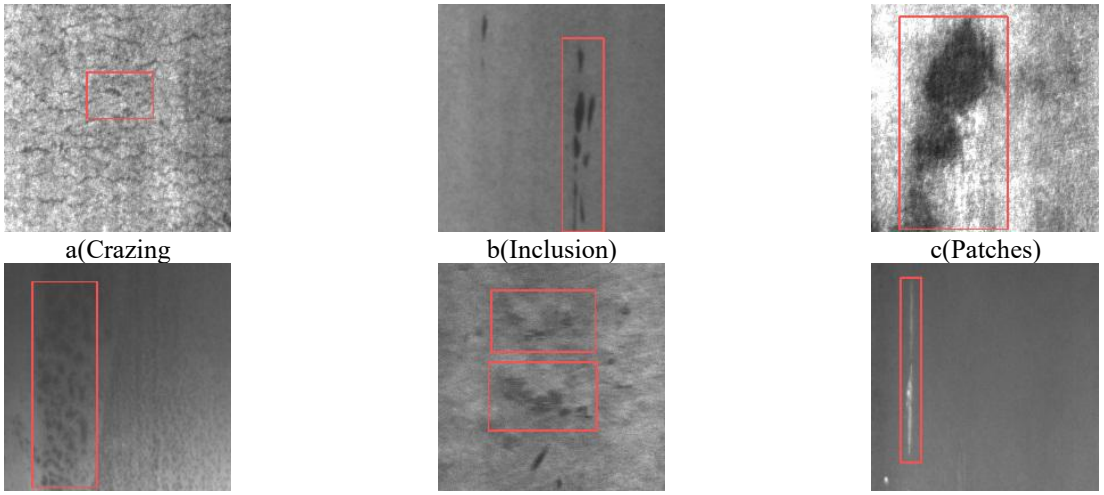



Fig. 5 Six steel surface defects in the NEU-DET dataset

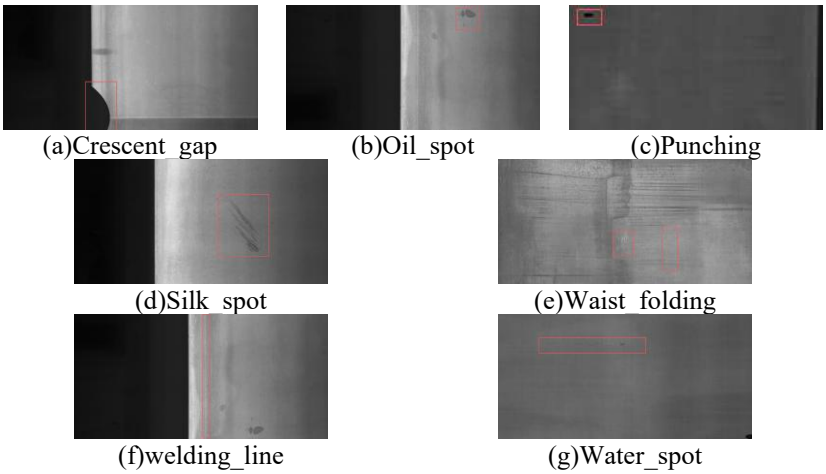



Fig. 6 Six surface defects on steel in the GC10-DET dataset

Table 2 Comparison of NEU-DET with the latest methods

| Methods | Params | GFLOPs | Cr | In | Pa | PS | RS | Sr | Recall | F1 | mAP@0.5 | mAP@0.75 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FasterR-CNN | 42.33 | 132.5 | 0.453 | 0.829 | 0.889 | 0.882 | 0.562 | 0.912 | 0.910 | 0.814 | 0.7545 | 0.372 |
| RetinaNet | 64.43 | 181.5 | 0.572 | 0.865 | 0.882 | 0.896 | 0.593 | 0.927 | 0.937 | 0.859 | 0.7892 | 0.401 |
| YOLOv8-n | 3.2 | 8.7 | 0.561 | 0.835 | 0.881 | 0.74 | 0.698 | 0.948 | 0.7552 | 0.7377 | 0.7771 | 0.4058 |
| YOLOv8-S | 11.2 | 28.6 | 0.511 | 0.863 | 0.906 | 0.836 | 0.65 | 0.928 | 0.7661 | 0.7613 | 0.7824 | 0.4150 |
| YOLOv8-m | 25.9 | 78.9 | 0.448 | 0.835 | 0.924 | 0.768 | 0.656 | 0.956 | 0.7401 | 0.7284 | 0.7644 | 0.3608 |
| YOLOv11-n | 4.2 | 10.5 | 0.332 | 0.731 | 0.854 | 0.641 | 0.618 | 0.93 | 0.6325 | 0.6417 | 0.6842 | 0.2881 |
| YOLOv11-s | 13.7 | 34.8 | 0.471 | 0.801 | 0.924 | 0.743 | 0.664 | 0.952 | 0.7130 | 0.7117 | 0.7592 | 0.3593 |
| YOLOv11-m | 29.2 | 95.6 | 0.428 | 0.761 | 0.9 | 0.728 | 0.702 | 0.936 | 0.7033 | 0.7075 | 0.7424 | 0.3582 |
| YOLOv12-n | 4.5 | 10.1 | 0.489 | 0.789 | 0.907 | 0.757 | 0.707 | 0.941 | 0.7084 | 0.7141 | 0.7650 | 0.3937 |
| YOLOv12-s | 15.2 | 34.9 | 0.601 | 0.886 | 0.874 | 0.769 | 0.603 | 0.96 | 0.7039 | 0.7578 | 0.7823 | 0.4218 |
| YOLOv12-m | 32.5 | 94.9 | 0.483 | 0.853 | 0.891 | 0.805 | 0.713 | 0.954 | 0.6992 | 0.7513 | 0.7831 | 0.4032 |
| Fous-DETR | 50.1 | 114 | 0.563 | 0.839 | 0.945 | 0.817 | 0.643 | 0.961 | 0.978 | 0.882 | 0.795 | 0.795 |
| FC-DETR | 57.8 | 142.5 | 0.682 | 0.841 | 0.921 | 0.891 | 0.714 | 0.956 | 0.891 | 0.889 | 0.834 | 0.502 |
| Es-NET | 148.9 | - | 0.553 | 0.883 | 0.879 | 0.865 | 0.591 | 0.951 | - | - | 0.7947 | - |
| CABF-FCOS | 57.4 | - | 0.561 | 0.749 | 0.941 | 0.878 | 0.619 | 0.845 | - | - | 0.766 | - |
| MTF-YOLO | 35 | 89 | 0.721 | 0.886 | 0.935 | 0.833 | 0.712 | 0.957 | 0.801 | 0.7628 | **0.841** | 0.4251 |

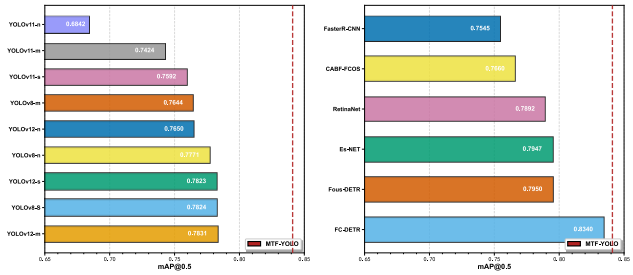



Fig. 7 Visual Comparison of MTF-YOLO with Other Methods

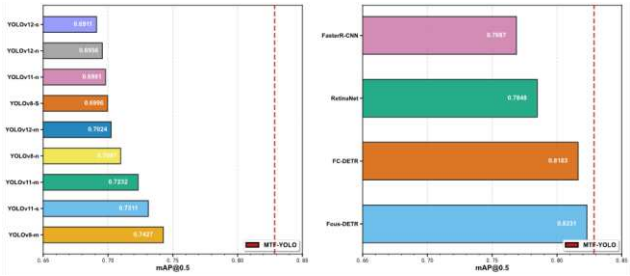



Fig. 8 Visualization Results of MTF-YOLO Compared to Other Methods

Table 3 Comparison of GC10-NET with the latest methods.

| Methods | Cg | Os | Pu | Ss | Wf | Ws | Wl | Recal | F1 | mAP@0.5 | mAP@0.75 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **FasterR-CNN** | 0.928 | 0.608 | 0.946 | 0.512 | 0.751 | 0.789 | 0.849 | 0.891 | 0.815 | 0.7687 | 0.302 |
| **RetinaNet** | 0.942 | 0.591 | 0.952 | 0.531 | 0.781 | 0.784 | 0.912 | 0.812 | 0.342 | 0.7848 | 0.334 |
| **YOLOv8-n** | 0.955 | 0.584 | 0.681 | 0.539 | 0.836 | 0.643 | 0.634 | 0.6436 | 0.6883 | 0.7097 | 0.3246 |
| **YOLOv8-S** | 0.975 | 0.639 | 0.713 | 0.56 | 0.846 | 0.613 | 0.551 | 0.6647 | 0.7025 | 0.6996 | 0.3232 |
| **YOLOv8-m** | 0.954 | 0.714 | 0.734 | 0.588 | 0.932 | 0.687 | 0.59 | 0.6695 | 0.7257 | 0.7427 | 0.3805 |
| **YOLOv11-n** | 0.958 | 0.675 | 0.669 | 0.501 | 0.831 | 0.664 | 0.589 | 0.6253 | 0.6704 | 0.6981 | 0.3309 |
| **YOLOv11-s** | 0.945 | 0.677 | 0.717 | 0.565 | 0.92 | 0.638 | 0.655 | 0.7194 | 0.7239 | 0.7311 | 0.3301 |
| **YOLOv11-m** | 0.965 | 0.687 | 0.73 | 0.608 | 0.82 | 0.631 | 0.621 | 0.6759 | 0.7021 | 0.7232 | 0.3499 |
| **YOLOv12-n** | 0.93 | 0.587 | 0.737 | 0.558 | 0.901 | 0.628 | 0.53 | 0.6448 | 0.6763 | 0.6956 | 0.2980 |
| **YOLOv12-s** | 0.95 | 0.63 | 0.73 | 0.523 | 0.897 | 0.596 | 0.512 | 0.6394 | 0.6760 | 0.6911 | 0.3402 |
| **YOLOv12-m** | 0.958 | 0.597 | 0.753 | 0.56 | 0.87 | 0.651 | 0.527 | 0.6818 | 0.6964 | 0.7024 | 0.3438 |
| **Fous-DETR** | 0.959 | 0.619 | 0.921 | 0.631 | 0.861 | 0.789 | 0.982 | 0.891 | 0.864 | 0.8231 | 0.429 |
| **FC-DETR** | 0.961 | 0.671 | 0.962 | 0.519 | 0.821 | 0.812 | 0.974 | 0.953 | 0.893 | 0.8163 | 0.461 |
| **MTF-YOLO** | 0.963 | 0.661 | 0.971 | 0.581 | 0.837 | 0.821 | 0.966 | 0.753 | 0.812 | **0.8286** | 0.4021 |

Table 4 Results Table for Each Module

| Case | Baseline | DVG | DEE | DOTF | map@0.5 |
|---|---|---|---|---|---|
| a | √ | - | - | - | 0.7545 |
| b | √ | √ | - | - | 0.8136 |
| c | √ | - | √ | - | 0.8175 |
| d | √ | - | - | √ | 0.8212 |
| e | √ | √ | √ | - | 0.8311 |
| f | √ | √ | - | √ | 0.8289 |
| g | √ | - | √ | √ | 0.8331 |
| h | √ | √ | √ | √ | 0.8410 |

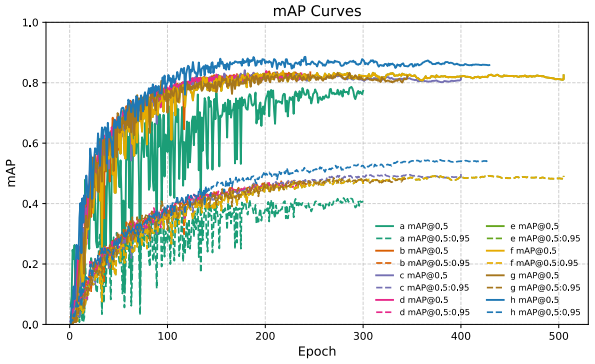



Fig. 9 Results of Adding Each Module

## 4.5 Validity of Each Module

### 4.5.1 Impact of DVGC on the Model

Fig. 10 plots P–R curves for the baseline and the +DVGC model. The +DVGC curve lies above the baseline, indicating a better precision–recall trade-off. At the highlighted points, the baseline achieves (P=0.753, R=0.755), while +DVGC achieves (P=0.818, R=0.799), demonstrating improved precision and recall. The annotation box reports summary metrics: baseline (mAP@0.5 = 0.786, precision = 0.753, recall = 0.755) and +DVGC (mAP@0.5 = 0.833, precision = 0.818, recall = 0.799).

Fig. 11 shows the density map of the Top-K candidate space for the DVGC case; brighter colors indicate higher normalized density. This single-panel figure presents only the DVGC case. Baseline comparisons and distance-to-boundary statistics are reported in the ablation table/appendix.

### 4.5.2 Impact of DEE on the Mode

To validate DEE, this paper conducted a controlled ablation in which only this layer was replaced, while the training strategy and hyperparameters were held constant. Fig. 12 plots recall–confidence curves for the DEE variant across defect classes (crazing, inclusion, patches, pitted_surface, rolled-in_scale, and scratches)

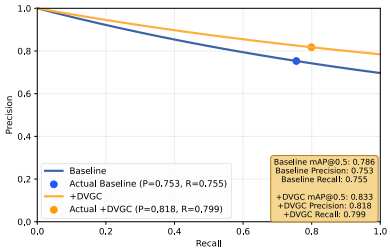



Fig. 10 P-R Curve

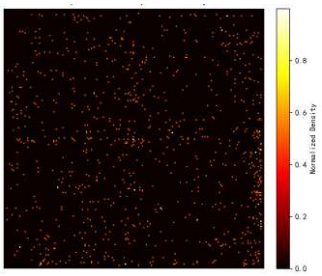


Fig. 11 Top-K Candidate Space

together with the aggregated "all classes" curve. Fig. 13 presents qualitative detections on the validation set (mainly scratches), illustrating well-localized boundaries and stable endpoints/corners. Quantitative comparisons to the baseline (e.g., recall and false-positive rates) are reported in the ablation table/appendix and are not displayed in these figures.

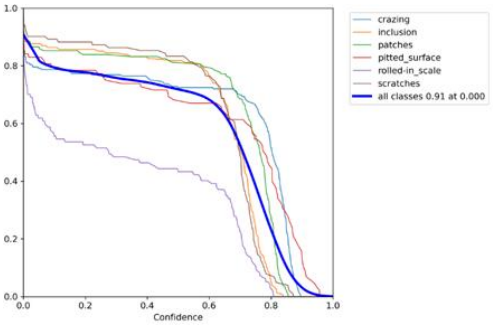



Fig. 12 R_curv

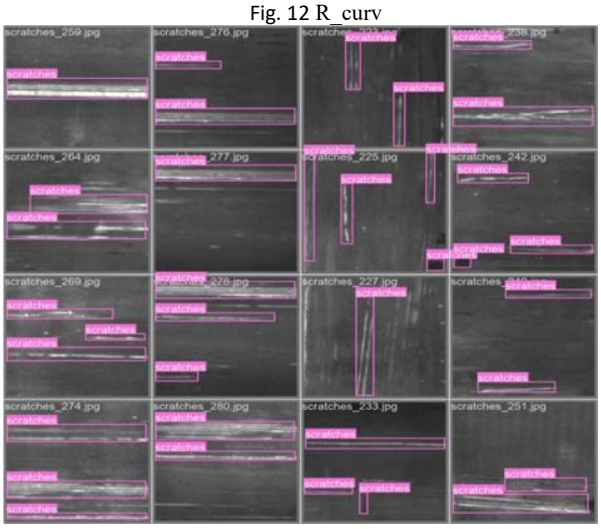


Fig. 13 Visualization of Validation Set Predictions

### 4.5.3 Impact of DOTF on the Model

The training results with DOTF are shown in Fig. 14. The top-left panel traces the precision–recall trajectory across epochs and marks the best epoch (216; ΔmAP = +0.470). The top-right panel reports the best-epoch metrics: mAP@0.5 = 0.832, mAP@0.5:0.95 = 0.476, precision = 0.856, and recall = 0.745. The bottom panel plots mAP@0.5 and mAP@0.5:0.95 over training epochs, showing steady increases and convergence toward the end. These plots characterize the training behavior and best-epoch performance of the DOTF variant. Baseline comparisons and latency/architectural details are provided elsewhere.

## 5.Conclusions

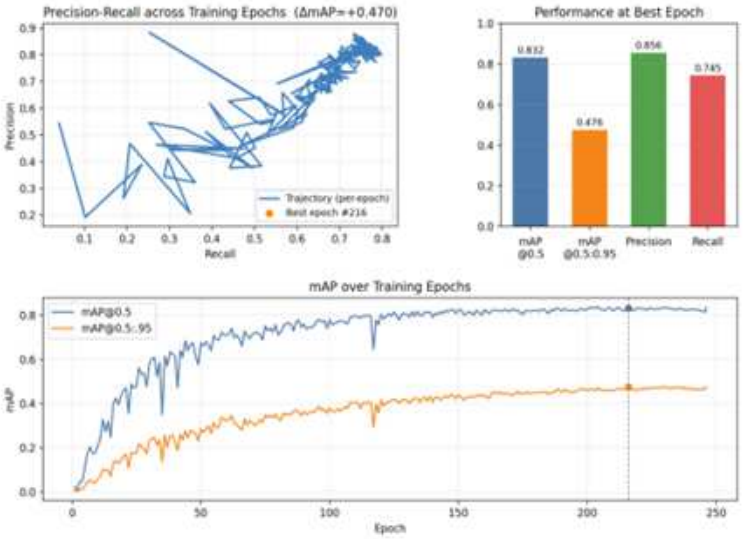



Fig. 14 Summarizes Training with DOTF

This paper addresses the challenge of detecting fine, elongated, low-contrast surface cracks on steel plates that are easily confused with grain patterns. Building on YOLOv8-n, this paper proposes a lightweight detector that incorporates crack-oriented stripe priors. Specifically, this paper inserts a Direction-Selective Stripe Fusion (DOTF) module at the neck to enhance sub-pixel linear structures while suppressing large-scale texture noise; DOTF uses eight orientations. This paper also integrates defect-variance guidance and edge enhancement into the backbone and neck to improve the separability and localization of weakly textured cracks, while maintaining comparable inference latency. Under a unified evaluation protocol, this paper reports AP@0.5, AP@0.75, recall, and F1, these metrics show consistent gains over strong YOLO baselines.

Future work will systematically evaluate discriminative losses and post-NMS refinement and stitching; introduce adaptive contrast enhancement and learnable normalization; combine semi-supervised self-training with structural distillation for cross-material generalization; and explore direction-consistent NMS, boundary-energy regression, scale-consistent stitching and deduplication, as well as inference-side quantization and plugin-based acceleration. These efforts aim to deliver an efficient solution deployable at the edge.

**Funding:** This research did not receive any financial support.

## References

[1]. Huo S, Zhang B, Muddassir M, et al. A sensor-based robotic line scan system with adaptive ROI for inspection of defects over convex free-form specular surfaces[J]. IEEE Sensors Journal, 2021, 22(3): 2782-2792.

[2]. He Y, Song K, Meng Q, et al. An end-to-end steel surface defect detection approach via fusing multiple hierarchical features[J]. IEEE transactions on instrumentation and measurement, 2019, 69(4): 1493-1504.

[3]. Zhang K, Li J, Li Q, et al. Improvement on electrochemical performance by electrodeposition of polyaniline nanowires at the top end of sulfur electrode[J]. Applied surface science, 2013, 285: 900-906.

[4]. Chen Y, Ding Y, Zhao F, et al. Surface defect detection methods for industrial products: A review[J]. Applied Sciences, 2021, 11(16): 7657.

[5]. LeCun Y, Bengio Y, Hinton G. Deep learning[J]. nature, 2015, 521(7553): 436-444.

[6]. Reis D, Kupec J, Hong J, et al. Real-time flying object detection with YOLOv8[J]. arXiv preprint arXiv:2305.09972, 2023.

[7]. Lin T Y, Dollár P, Girshick R, et al. Feature pyramid networks for object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 2117-2125.

[8]. Dosovitskiy A. An image is worth 16x16 words: Transformers for image recognition at scale[J]. arXiv preprint arXiv:2010.11929, 2020.

[9]. Carion N, Massa F, Synnaeve G, et al. End-to-end object detection with transformers[C]//European conference on computer vision. Cham: Springer International Publishing, 2020: 213-229.

[10]. Zhu X, Su W, Lu L, et al. Deformable detr: Deformable transformers for end-to-end object detection[J]. arXiv preprint arXiv:2010.04159, 2020.

[11]. Bochkovskiy A, Wang C Y, Liao H Y M. Yolov4: Optimal speed and accuracy of object detection[J]. arXiv preprint arXiv:2004.10934, 2020.

[12]. Reis D, Kupec J, Hong J, et al. Real-time flying object detection with YOLOv8[J]. arXiv preprint arXiv:2305.09972, 2023.

[13]. Gonzalez R C. Digital image processing[M]. Pearson education india, 2009.

[14]. Hu J, Shen L, Sun G. Squeeze-and-excitation networks[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 7132-7141.

[15]. Lin T Y, Dollár P, Girshick R, et al. Feature pyramid networks for object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 2117-2125.

[16]. Goodfellow I, Bengio Y, Courville A, et al. Deep learning[M]. Cambridge: MIT press, 2016.

[17]. Sobel I. An isotropic 3× 3 image gradient operater[J]. Machine vision for three-dimensional scenes, 1990: 376-379.

[18]. Dai J, Qi H, Xiong Y, et al. Deformable convolutional networks[C]//Proceedings of the IEEE international conference on computer vision. 2017: 764-773.

[19]. Chen X, Kundu K, Zhu Y, et al. 3d object proposals for accurate object class detection[J]. Advances in neural information processing systems, 2015, 28.

[20]. Liu S, Qi L, Qin H, et al. Path aggregation network for instance segmentation[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 8759-8768.

[21]. Woo S, Park J, Lee J Y, et al. Cbam: Convolutional block attention module[C]//Proceedings of the European conference on computer vision (ECCV). 2018: 3-19.

[22]. Daugman J G. Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters[J]. Journal of the Optical Society of America A, 1985, 2(7): 1160-1169.

[23]. Dalal N, Triggs B. Histograms of oriented gradients for human detection[C]//2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05). Ieee, 2005, 1: 886-893.

[24]. He Y, Song K, Meng Q, et al. An end-to-end steel surface defect detection approach via fusing multiple hierarchical features[J]. IEEE transactions on instrumentation and measurement, 2019, 69(4): 1493-1504.

[25]. Ge Z, Liu S, Wang F, et al. Yolox: Exceeding yolo series in 2021[J]. arXiv preprint arXiv:2107.08430, 2021.