FastFlow-DETR: An Efficient Steel Surface Defect Detection Algorithm Based on RT-DETR

Yihan Chai[1], Qiming Yu[2,*], and Chengming Liu[1,*]
[1]School of Cyber Science and Engineering, Zhengzhou University, Zhengzhou, China
[2]School of Information Science and Technology, Zhengzhou Normal University, Zhengzhou, China
*Corresponding authors: qmy.09@163.com, cmliu@zzu.edu.cn

***Abstract*—Surface defect detection in steel is critical for industrial quality control. However, existing detection frameworks tend to dilute fine-grained defect information during multi-level feature processing and introduce unnecessary computational overhead. This leads to missed detection of weak defects, unstable performance, and limited real-time applicability in industrial scenarios. In this paper, we introduce FastFlow-DETR, a lightweight and efficient framework built upon RT-DETR, designed to address these challenges. The backbone incorporates a Selective Spatial Block (SSB) to preserve critical local features, while a Branch Interweave Module (BIM) is used during feature fusion to align shallow detail features with deep semantic representations. In addition, a Dual Dynamic Multi-Scale Relay (D^2MR) encoder is proposed to enhance feature interaction across different representation levels, within which a Self-Scaled Tanh Normalization (STN) stabilizes feature modulation and prevents excessive suppression of weak defect responses. Experimental results on GC10-DET and NEU-DET show that FastFlow-DETR achieves higher detection accuracy than RT-DETR, with mAP@0.5 gains of 2.9 and 2.4 percentage points, respectively. Meanwhile, the proposed method reduces computational complexity by 32.1% in FLOPs and model parameters by 29.4%. These results show that the proposed method effectively balances detection accuracy and computational efficiency for industrial steel surface defect detection.**

***Index Terms*—steel surface defect detection, RT-DETR, lightweight detection, multi-scale attention**

I. Introduction

Steel surface defect detection is important for industrial quality inspection, as surface defects directly affect the reliability and service safety of steel products [1]. During rolling and manufacturing, steel plates often contain defects such as cracks, scratches, pits, and oxide scales. These defects vary in scale, shape, and texture. Many appear as thin structures or weak-contrast patterns, which makes accurate detection difficult. In practical inspection, existing methods struggle to preserve fine-grained defect information while maintaining real-time efficiency. This leads to limited detection accuracy and unstable performance.

With the rapid development of deep learning, convolutional neural networks (CNNs) have become the mainstream solution for automated surface defect detection. Representative detectors such as Faster R-CNN [2], RetinaNet [3], and YOLO series models [4]–[10], achieve strong performance by learning hierarchical feature representations from data. Several studies have further adapted CNN-based detectors for steel inspection by enhancing feature fusion or attention mechanisms [11], [12]. Despite their success, CNN-based methods still face limitations in real industrial environments. Strong reflections, complex textures, and background noise often interfere with feature extraction, leading to false positives or missed detections. Moreover, industrial inspection systems usually impose strict constraints on inference speed and computational cost, which restricts the deployment of heavy models.

Transformer-based detectors have recently attracted increasing attention due to their strong global modeling capability. DETR introduces an end-to-end detection paradigm by formulating object detection as a set prediction problem, eliminating handcrafted components such as anchors and non-maximum suppression [13].

More recently, RT-DETR further optimizes the encoder design and enables real-time object detection while retaining transformer-style global reasoning [14]. However, when directly applied to steel surface defect detection, RT-DETR still exhibits several limitations. First, its backbone performs spatial computation uniformly across channels, introducing redundant operations while weakening responses to localized defect patterns. Second, under strict efficiency constraints, the encoder lacks sufficient multi-scale interaction, causing small or weak defects to be gradually diluted during deep encoding. Third, feature fusion between shallow detail features and deep semantic representations is not explicitly aligned, which degrades detection performance for irregular and low-contrast defects commonly observed on steel surfaces.

To address these challenges, we propose FastFlow-DETR, a lightweight and efficient steel surface defect detection framework built upon RT-DETR.

The main contributions of this work are summarized as follows:

- Selective Spatial Block (SSB) is proposed to address redundant spatial computation in the backbone. It selectively enhances local defect-sensitive regions and improves the representation of small and weak defects with lower computational cost.
- Branch Interweave Module (BIM) is designed for feature fusion. It enables explicit interaction and alignment between shallow detail features and deep semantic features. This reduces feature misalignment caused by scale inconsistency.
- Dual Dynamic Multi-Scale Relay (D^2MR) module is introduced to strengthen multi-scale feature interaction

in the encoder. It improves the preservation of small-scale defect information under real-time computational constraints.

II. Related Work

A. Traditional vision-based methods

Early steel surface defect detection mainly relied on classical machine vision and image processing techniques. These methods usually extract handcrafted features, such as texture responses, edge information, or intensity statistics, followed by rule-based decision strategies. Choi et al. [15] proposed a pinhole defect detection method for steel slabs based on Gabor filter combinations, which enhances texture contrast to identify defects. In parallel, machine vision–based image processing techniques have been extensively applied for surface finish assessment and defect inspection in manufacturing processes, typically using edge detection and histogram-based feature analysis [16].

Although traditional methods are computationally efficient, their performance strongly depends on manually designed features and imaging conditions. They are sensitive to illumination changes, surface reflections, and background noise. This limits their robustness and generalization ability in complex industrial environments.

B. Deep learning-based defect detection methods and challenges

With the development of deep learning, CNN-based detectors have been widely applied to steel surface defect detection [17]. Du et al. [12] proposed AFF-Net, which introduces adaptive focusing to improve defect detection on strip steel surfaces. Shi et al. [18] developed GCF-Net by integrating global attention and inter-layer feature interaction. Yu et al. [11] designed an anchor-free detector that combines channel attention with bidirectional feature fusion, achieving improved performance on steel strip datasets. However, under real-time industrial constraints, CNN-based methods still struggle to preserve fine-grained defect features at low computational cost.

Recent studies have explored hybrid architectures for industrial surface inspection. Song et al. [19] improved YOLOv7 by integrating the SAM-SPPFCSPC structure with depthwise separable convolutions. Yang and Liu [20] enhanced RetinaNet by introducing CA-BiFPN and IA-BCELoss to improve small defect detection. Chen et al. [21] proposed HCT-Det, which balances detection accuracy and computational efficiency using the WSA-R module. Zhou et al. [22] introduced a multi-scale attention mechanism into YOLOv8 to improve detection of low-contrast defects. Lu et al. [23] improved convolutional feature fusion in the backbone network to enhance detection efficiency. Despite these advances, existing detectors still struggle to preserve fine-grained features and build efficient multi-scale representations.

To enhance global feature modeling, transformer-based detectors have attracted increasing attention. Subsequent variants, including Deformable DETR, improve convergence and efficiency by introducing sparse attention mechanisms [24]. RT-DETR further adopts hybrid encoders and IoU-aware query selection to achieve real-time detection [14]. When applied to industrial steel surface defect inspection, RT-DETR still shows limitations. Its AIFI encoder provides limited multi-scale feature interaction. The feed-forward process tends to smooth local textures. Fine-grained details are gradually lost. In addition, redundant channel computation and dense attention reduce efficiency. These factors make it difficult to accurately detect small and low-contrast defects under strict real-time constraints.

Based on these observations, this paper proposes FastFlow-DETR, a redesigned RT-DETR framework tailored for industrial steel surface defect detection. The proposed method achieves a better balance between detection accuracy, robustness, and computational efficiency.

III. Methodology

A. Overview of FastFlow-DETR

To address the challenges of detecting diverse, irregular, and fine-grained surface defects on steel plates, we redesign the RT-DETR architecture and propose an efficient detection framework named FastFlow-DETR. The proposed framework aims to improve feature representation for small-scale and low-contrast defects while maintaining real-time inference efficiency.

As illustrated in Fig. 1, FastFlow-DETR introduces a progressive feature interaction mechanism across the backbone, encoder, and fusion stages. Instead of processing semantic and detailed information in a uniform manner, FastFlow-DETR enables information to flow sequentially through multiple modules. This design forms a structured semantic-detail relay that supports continuous feature refinement across different network depths. Compared with conventional end-to-end detectors, FastFlow-DETR explicitly enhances the interaction between local spatial details and high-level semantic features. The overall architecture is carefully optimized to reduce redundant computation, making FastFlow-DETR suitable for real-time industrial defect inspection.

B. SSB for Efficient Backbone Encoding

We replace the original BasicBlock in the backbone network with the proposed SSB, which is designed to enhance local defect-sensitive features while reducing redundant spatial computation. The overall workflow of SSB is illustrated in Fig. 2.

SSB is embedded into the residual structure of the original BasicBlock. The input feature first passes through a standard 3×3 convolution to obtain an initial spatial representation. The feature is then processed by a selective spatial convolution core, inspired by partial convolution (PConv) techniques [25], where the channel dimension is split and only a subset of channels undergoes spatial convolution. This selective approach reduces redundant computations and focuses feature extraction on the most relevant regions, enhancing defect-sensitive responses.

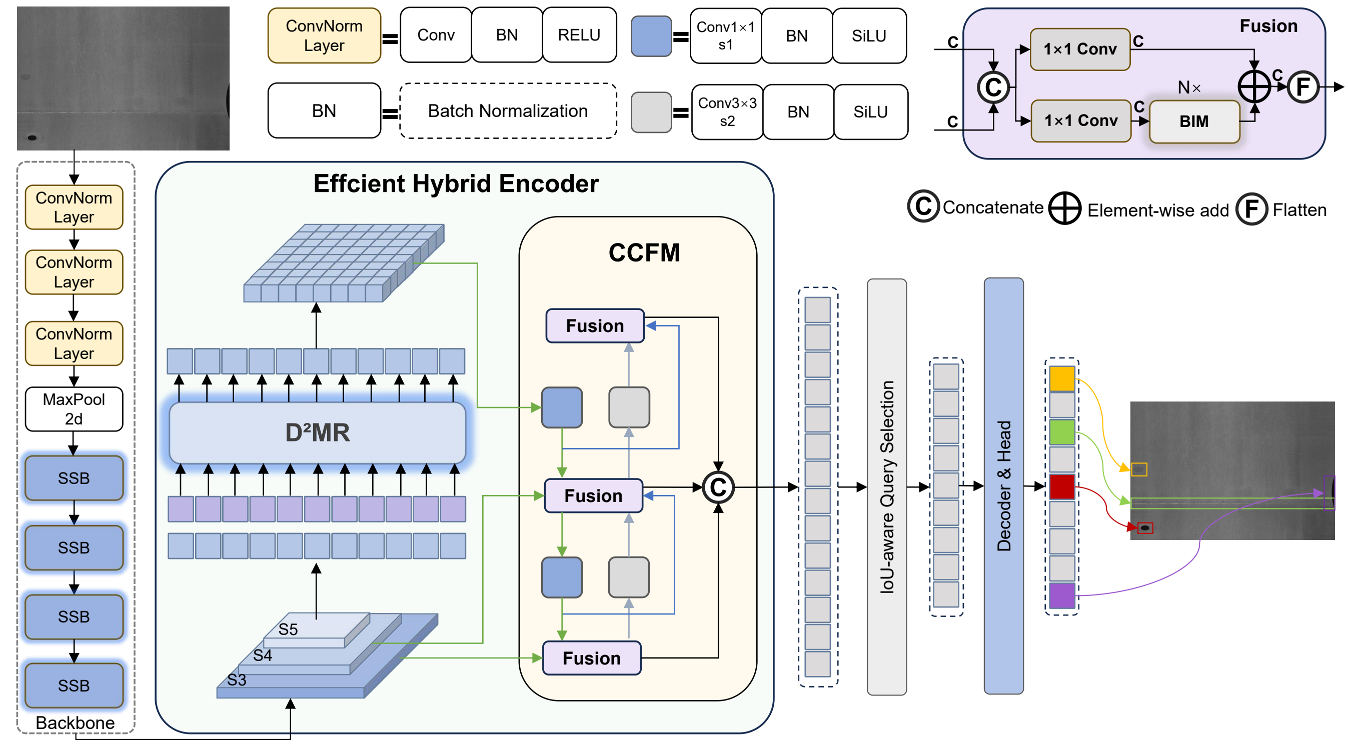

Fig. 1. Overall architecture of the proposed FastFlow-DETR. The framework is composed of an SSB-enhanced backbone, a BIM-based feature fusion stage, and a D²MR encoder, forming a structured semantic–detail relay from shallow spatial details to deep semantic representations.

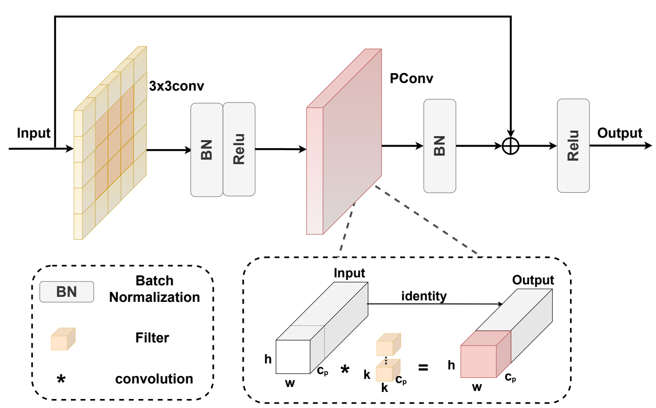

Fig. 2. The structure of SSB.

For an input $x \in \mathbb{R}^{B\times C\times H\times W}$, the channel dimension is partitioned as $x = [x_1, x_2]$, where $x_1 \in \mathbb{R}^{B\times C_1\times H\times W}$ and $x_2 \in \mathbb{R}^{B\times C_2\times H\times W}$, with $C = C_1 + C_2$. The partial branch computes:

$$y_1 = \mathrm{Conv}_{3\times 3}(x_1), \quad y_2 = x_2 \tag{1}$$

where y_1 captures local spatial information and y_2 preserves identity features.

To maintain dimensional consistency, a lightweight 1×1 projection of the input is adopted as the residual shortcut. The final output of the block is given by:

$$\mathrm{out} = \mathrm{ReLU}\big(\mathrm{BN}(\mathrm{Concat}(\mathrm{Conv}_{3\times 3}(x_1), x_2)) + \mathrm{Proj}_{1\times 1}(x)\big) \tag{2}$$

By selectively applying spatial convolution on partial channels, SSB preserves fine-grained local details that are critical for small and low-contrast defects. At the same time, redundant computation on less informative channels is avoided, making SSB efficient and suitable for deep residual stacking in FastFlow-DETR.

C. BIM for Feature Fusion

In the feature fusion stage of RT-DETR [14], the original RepBlock aggregates multi-level features in a largely uniform manner. Under complex steel surface textures, this may lead to insufficient alignment between shallow detail cues and deep semantic representations. To address this limitation, we replace RepBlock with the proposed BIM, as illustrated in Fig. 3.

BIM follows a three-branch interweaving design and introduces a lightweight Response Modulator (RM) in the middle branch to explicitly regulate feature contributions during fusion. By coordinating high-frequency structures, weak response cues, and background priors, BIM improves fusion consistency while keeping the module efficient.

The computation begins with a 1×1 convolution that expands the input features:

$$f = \mathrm{Conv}_{1\times 1}(x), \quad f \in \mathbb{R}^{B\times 2c\times H\times W} \tag{3}$$

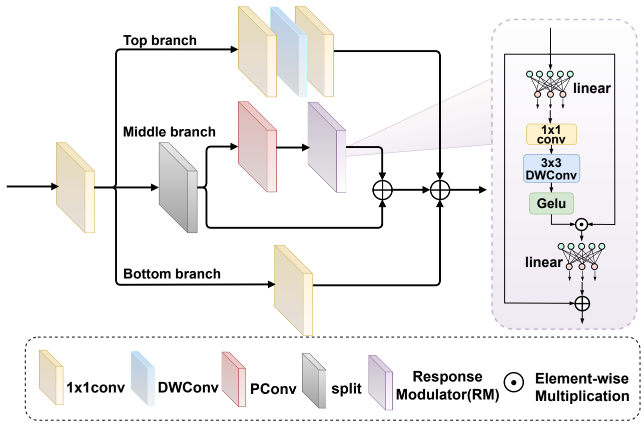

Fig. 3. The structure of BIM.

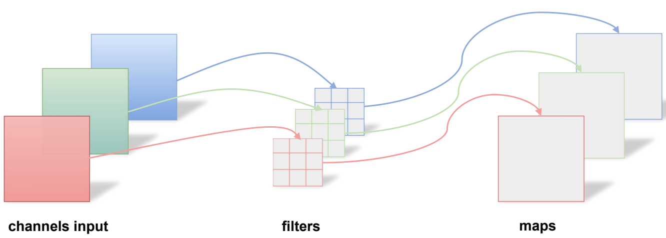

Fig. 4. Illustration of depthwise convolution.

This projection increases channel diversity and provides sufficient capacity for branch-specific processing.

1) Top branch: The top branch focuses on capturing fine local structures such as thin scratches and crack edges. Applying dense spatial convolution across all channels is computationally redundant for smooth background regions. Therefore, as shown in Fig. 4, an efficient depthwise convolution [26] to perform channel-wise spatial filtering:

$$t = W_t^{(2)}\left(\mathrm{DWConv}_{3\times3}(W_t^{(1)}(f_t))\right) \tag{4}$$

This design preserves edge-sensitive responses while significantly reducing computational overhead.

2) Middle branch: The middle branch is designed to enhance weak and small-scale responses that are easily suppressed during feature fusion. The middle branch first applies selective spatial processing via partial convolution and then uses the proposed RM to reshape intermediate responses before fusion.

After the intermediate feature is produced, it is split into two parts $m = [m_a, m_b]$. One part generates a modulation signal through depthwise convolution and nonlinear activation:

$$g = \mathrm{GELU}(\mathrm{DWConv}_{3\times3}(m_a)) \tag{5}$$

and the other part is modulated by element-wise multiplication and refined with a residual connection:

$$\tilde{m} = W_m^{(1)}(m_b \odot g) + m_b \tag{6}$$

Here $\odot$ denotes element-wise multiplication. The residual term preserves the original response pathway and stabilizes feature propagation.

For clarity, the RM operation can be summarized as:

$$\tilde{m} = m_b \odot \phi(m_a) + m_b \tag{7}$$

where $\phi(\cdot)$ denotes the modulation function implemented by DWConv and GELU. This response shaping suppresses background-dominant activations and strengthens informative weak cues without introducing heavy attention computation.

3) Bottom branch: The bottom branch provides a low-cost pathway to preserve background priors and low-frequency information. A lightweight pointwise projection is employed to maintain stable representations and reduce over-response in homogeneous regions.

Finally, the outputs of the three branches are concatenated and fused through residual interaction. The top branch strengthens high-frequency structural cues, the middle branch reshapes weak responses, and the bottom branch stabilizes background-aware features. This interweaving fusion strategy improves semantic–detail alignment and reduces missed detections of small or low-contrast defects, while keeping the overall module lightweight.

D. D²MR Encoder for Progressive Feature Interaction

The AIFI encoder in RT-DETR [14] is effective for modeling global dependencies, but it provides limited multi-scale interaction under real-time constraints. As a result, small and weak defects are often suppressed by dominant background responses. To address this issue, we propose the D^{2}MR encoder to replace the original AIFI module. D^{2}MR adopts a relay-style design that progressively refines semantic and detail information through two complementary modules, namely MDAB and MGDB. The overall architecture is shown in Fig. 5.

1) Multi-scale Dynamic Attention Block (MDAB): The MDAB is a sparse and dense multi-scale attention mechanism that enables each query to dynamically select between local and global candidate tokens. This design enables the model to focus on minute defects while maintaining an awareness of larger defects.

Specifically, given an input feature $X \in \mathbb{R}^{B\times C\times H\times W}$, the queries Q, keys K, and values V are obtained through linear projections:

$$(Q, K, V) = (W_q X,\ W_k X,\ W_v X) \tag{8}$$

We then construct two types of attention: dense attention to ensure global dependencies and sparse attention to highlight local strong responses:

$$A_{\text{dense}} = \mathrm{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right) \tag{9}$$

$$A_{\text{sparse}} = \mathrm{ReLU}^2\left(QK^T\right) \tag{10}$$

These attention maps are fused to produce a dynamic attention

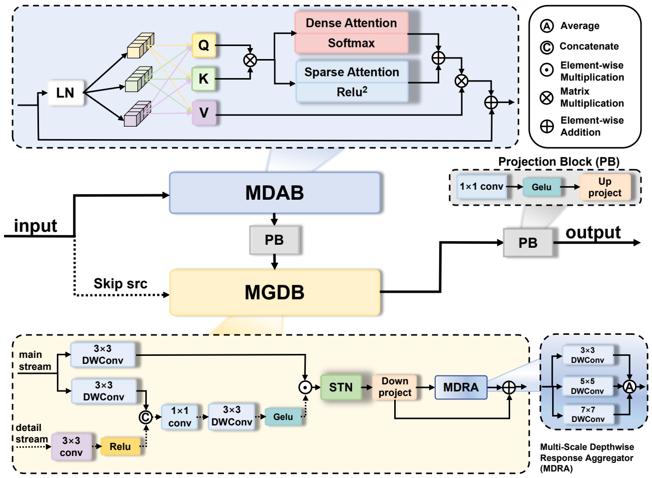

Fig. 5. The structure D^{2}MR.

representation:

$$F_{\text{att}} = (A_{\text{dense}} + A_{\text{sparse}})V \tag{11}$$

By combining dense global modeling with sparse local enhancement, MDAB ensures that small defects are amplified before deep feature propagation. This design enables effective information relay from shallow details to deeper semantic representations.

2) Self-Scaled Tanh Normalization (STN): To stabilize feature amplitudes and prevent weak defect signals from being overly suppressed, we introduce a lightweight normalization function termed STN. Given an input feature x, STN is defined as:

$$\text{STN}(x) = \tanh\left(\frac{x}{\mathbb{E}(|x|) + \epsilon}\right) \tag{12}$$

where $\mathbb{E}(|x|)$ denotes the mean absolute value of the input feature and ϵ is a small constant for numerical stability.

Unlike batch normalization or layer normalization, STN does not rely on mean subtraction or variance normalization. Instead, it rescales feature amplitudes in a data-dependent manner and applies bounded nonlinear compression. This design helps preserve weak and low-contrast defect responses while suppressing extreme activations caused by reflections or noise, which is particularly suitable for steel surface defect detection.

3) Multi-scale Gated Dual-stream Block (MGDB): The MGDB is designed to preserve fine-grained defect responses while maintaining global semantic stability. It adopts a dual-stream structure composed of a main stream and a detail stream.

Given an input feature F, it is first split into two branches. The main stream S_{main} retains global and low-frequency semantics, which describe background regions and large-scale defects. The detail stream S_{det} focuses on high-frequency structures, such as crack edges and small pits, through lightweight convolution and activation.

A gating mechanism is applied to adaptively modulate the contribution of detail features:

$$G = \sigma(WS_{\text{det}}) \tag{13}$$

and the gated fusion is computed as

$$F_{\text{gated}} = S_{\text{main}} + G \odot S_{\text{det}} \tag{14}$$

To avoid suppressing weak defect responses during nonlinear modulation, STN is applied:

$$F_{\text{norm}} = \text{STN}(F_{\text{gated}}) \tag{15}$$

Finally, a Multi-Scale Depthwise Response Aggregator (MDRA) is employed to re-balance spatial responses under different receptive fields:

$$F_{\text{MGDB}} = \sum_{k\in\{3,5,7\}} \text{DWConv}_{k\times k}(F_{\text{norm}}) \tag{16}$$

This design enables MGDB to preserve global context, enhance weak defect signals, and stabilize spatial responses across different defect scales.

Through this relay-style refinement, MDAB first highlights potential defect cues, while MGDB selectively strengthens fine details and suppresses background noise. Together, they form a progressive semantic-detail relay mechanism that improves robustness to scale variation and defect appearance diversity.

IV. Experiment and Result Analysis

A. Dataset and environment

To validate the effectiveness of the proposed method, two representative benchmark datasets for steel surface defect detection were selected: GC10-DET [27] and NEU-DET [28]. The GC10-DET dataset is one of the most widely adopted benchmarks in the steel industry. It contains a total of 2,294 high-resolution color images annotated with ten representative defect types, including Crescent (Cg), Inclusion (In), Rolled-in scale (Rp), Weld line (Wl), Crease (Cr), Silk spot (Ss), Oil spot (Os), Water spot (Ws), Punching (Pu), and Waist fold (Wf). Examples of annotated defect images are illustrated in Figure 6. The NEU-DET dataset, developed by Northeastern University, contains 1,800 grayscale images evenly divided into six defect categories: crazing (Cr), inclusion (In), pitted surface (Ps), rolled-in scale (Rs), patches (Pa), scratch (Sc). Each class contains 300 samples, ensuring a balanced distribution for comprehensive evaluation. Representative examples of NEU-DET defect images are shown in Figure 7.

The experiments were conducted using the PyTorch deep learning framework under a Linux environment. The training was performed on a single NVIDIA RTX 3080 Ti GPU with 12 GB of VRAM, utilizing CUDA 11.8 and Python 3.10. The batch size and number of workers were both set to 8, and training was performed without caching to ensure stable data loading. The SGD optimizer implemented in this study was employed, with a learning rate of 0.01, momentum of 0.9, and weight decay of 0.0005.

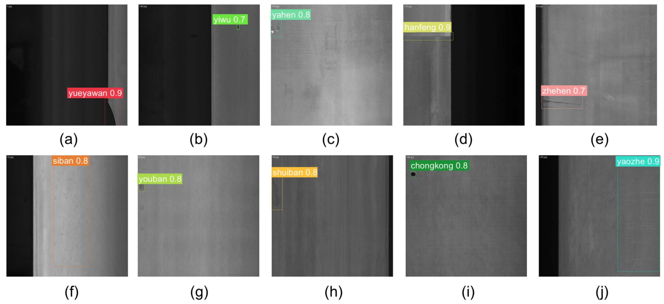

Fig. 6. Visualization of detection results on the GC10-DET dataset. (a) Crescent (Cg), (b) Inclusion (In), (c) Rolled-in scale (Rp), (d) Weld line (Wl), (e) Crease (Cr), (f) Silk spot (Ss), (g) Oil spot (Os), (h) Water spot (Ws), (i) Punching (Pu), (j) Waist fold (Wf).

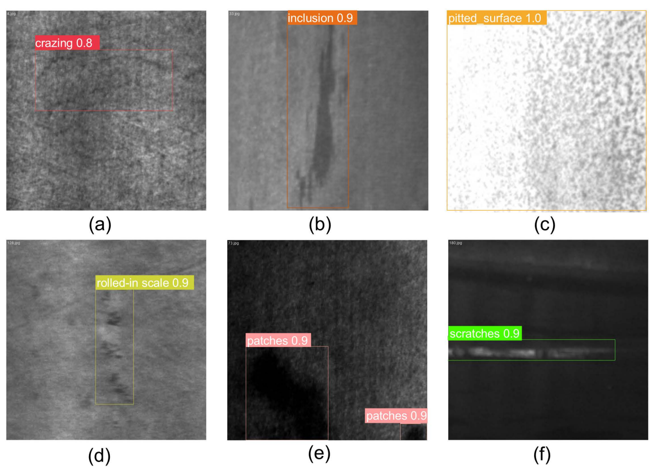

Fig. 7. Visualization of detection results on the NEU-DET dataset. (a) crazing (Cr), (b) inclusion (In), (c) pitted surface (Ps), (d) rolled-in scale (Rs), (e) patches (Pa), (f) scratch (Sc).

B. Evaluation Metrics

To comprehensively evaluate performance, the following metrics are employed:

Average Precision (AP): quantifies the average performance of the precision-recall curve for a single category. It is calculated as:

$$AP = \int_0^1 P(R)dR \tag{17}$$

where $P(R)$ represents the precision at a given recall R.

mean Average Precision (mAP): defined as

$$mAP = \frac{1}{n}\sum_{i=1}^{n} AP_i \tag{18}$$

where AP_i denotes the average precision of the i-th category and n is the number of categories.

FLOPs: number of floating-point operations per forward pass, measuring computational complexity.

Params: number of parameters, reflecting storage and deployment cost.

FPS (Frames Per Second) measures the inference speed of the detector during testing and reflects its real-time capability for industrial deployment.

C. Comparison with Existing Methods

1) Results on GC10-DET: On the GC10-DET dataset, as shown in Table I, FastFlow-DETR achieves the highest mAP@50 of 68.0%. It outperforms both two-stage detectors and recent real-time one-stage methods.

Compared with Faster R-CNN, FastFlow-DETR improves mAP@50 by 4.1%. The inference speed is more than five times higher, which supports real-time inspection. FastFlow-DETR also outperforms RetinaNet and FCOS, with mAP@50 improvements of 14.2% and 8.8%, respectively. Compared with RT-DETR, FastFlow-DETR achieves a 2.9% gain in mAP@50 while maintaining a high inference speed of 119.8 FPS. Some YOLO-based detectors achieve higher FPS, but their detection accuracy is lower. FastFlow-DETR provides a better balance between accuracy and speed. At the category level, FastFlow-DETR performs well on defects with strong structural characteristics. The AP values for Cg and Wl reach 95.4% and 91.0%, respectively. For elongated and weak-texture defects such as Os and Ws, the AP values reach 69.1% and 87.9%. The proposed method also performs well on defect types with weak contrast and fine textures. For Ss, Os, and Wf categories, FastFlow-DETR maintains stable detection accuracy under complex background conditions. On the Pu category, the AP reaches 95.5%, indicating improved localization of highly localized defects.

2) Results on NEU-DET: The quantitative results on the NEU-DET dataset are reported in Table II. FastFlow-DETR achieves the highest mAP@50 of 76.4%.

Compared with Faster R-CNN, FastFlow-DETR improves mAP@50 by 5.3% with significantly higher inference speed. FastFlow-DETR also outperforms single-stage detectors such as RetinaNet, FCOS, and ATSS, especially on small-scale and low-contrast defects. Compared with RT-DETR, FastFlow-DETR achieves a 2.4% gain in mAP@50 while maintaining comparable inference speed.

Among YOLO-based detectors, FastFlow-DETR outperforms YOLOv8 and YOLOv12 by 2.7% and 2.5% mAP@50, respectively. The inference speed remains high at 121.6 FPS. At the category level, the AP values for In and Pa reach 81.9% and 92.5%, respectively. For challenging categories such as Cr and Rs, which are sensitive to background noise and illumination variation, FastFlow-DETR maintains robust performance. The AP values reach 53.2% and 63.2%, respectively. The detection accuracy on Sc and Ps also remains competitive, reaching 87.6% and 79.9%. These results indicate stable detection performance under background noise and illumination variation.

D. Ablation Studies

To investigate the contribution of each component in FastFlow-DETR, ablation experiments are conducted on the GC10-DET and NEU-DET datasets. The quantitative results

TABLE I
COMPARISON RESULTS ON THE GC10-DET DATASET

Model	mAP@50(%)↑	FPS	AP@50(%)↑									
			Cg	In	Rp	Wl	Cr	Ss	Os	Ws	Pu	Wf
RT-DETR [14]	65.1	99.1	94.7	31.6	21.6	88.0	26.5	56.6	67.9	87.4	94.5	81.9
DETR [13]	61.9	85.2	86.2	28.6	35.8	69.1	**42.4**	59.0	60.8	67.3	89.4	79.9
Faster-RCNN [2]	63.9	23.1	85.6	30.5	**36.3**	72.3	41.8	**59.2**	64.9	70.1	94.3	83.3
CenterNet [29]	62.3	61.9	84.2	31.7	29.8	74.8	36.0	51.7	65.7	74.7	94.1	80.6
RetinaNet [3]	53.8	41.8	90.8	17.4	17.3	71.2	11.8	26.3	63.8	74.5	93.7	71.3
FCOS [30]	59.2	48.7	88.1	32.2	34.1	60.3	22.1	56.1	58.2	65.4	91.2	84.1
ATSS [31]	58.4	44.8	87.1	16.2	30.2	80.4	20.7	54.3	62.3	68.9	91.2	71.6
YOLOv5 [32]	63.2	80.6	94.4	30.5	24.2	77.6	17.5	57.3	67.0	84.3	97.8	80.9
YOLOv8 [6]	64.6	**137.4**	95.0	27.5	29.7	87.3	8.2	56.3	68.3	87.5	**98.4**	**87.9**
YOLOv9 [7]	64.2	99.4	94.0	31.4	27.6	88.9	13.8	54.8	66.9	85.2	97.2	82.5
YOLOv10 [8]	54.0	105.9	87.4	23.9	16.0	30.2	18.7	51.5	60.4	79.2	96.1	77.0
YOLOv11 [9]	55.7	112.7	92.2	25.7	18.2	32.1	3.8	55.4	64.9	86.2	97.0	81.9
YOLOv12 [10]	62.0	109.8	94.6	21.4	27.3	90.4	21.4	51.3	67.1	83.2	92.2	71.2
FastFlow-DETR	**68.0**	119.8	**95.4**	**32.3**	33.4	**91.0**	36.4	55.3	**69.1**	**87.9**	95.5	83.9

TABLE II
COMPARISON RESULTS ON THE NEU-DET DATASET

Model	mAP@50(%)↑	FPS	AP@50(%)↑					
			Cr	In	Pa	Ps	Rs	Sc
RT-DETR [14]	74.0	102.4	49.6	81.4	90.5	75.2	58.3	87.5
DETR [13]	72.1	88.6	45.7	75.4	89.6	82.1	58.9	80.4
Faster-RCNN [2]	71.1	17.8	42.9	67.9	84.9	79.1	**65.8**	86.1
CenterNet [29]	75.2	63.8	44.6	77.1	92.1	**86.4**	62.5	**88.7**
RetinaNet [3]	71.3	41.2	42.0	77.2	90.1	84.3	62.4	71.6
FCOS [30]	72.0	50.1	47.9	72.8	90.6	78.3	61.8	80.6
ATSS [31]	63.1	46.0	36.3	71.1	79.7	78.3	50.4	62.4
YOLOv5 [32]	71.7	82.9	46.8	76.3	91.5	84.0	56.4	75.1
YOLOv8 [6]	73.7	**140.2**	47.9	80.6	91.4	84.0	57.5	80.7
YOLOv9 [7]	70.6	101.8	36.2	74.1	91.0	83.3	56.5	82.2
YOLOv10 [8]	65.3	108.7	38.3	69.7	87.5	78.0	49.1	69.2
YOLOv11 [9]	75.1	114.9	50.0	77.6	92.4	84.4	62.8	83.3
YOLOv12 [10]	73.9	110.1	49.9	73.4	90.1	83.7	60.9	84.7
FastFlow-DETR	**76.4**	121.6	**53.2**	**81.9**	**92.5**	79.9	63.2	87.6

TABLE III
ABLATION STUDY ON NEU-DET AND GC10-DET DATASETS

Methods	mAP@50(%)↑		GFLOPs(G)↓	Params(M)↓
	GC10-DET	NEU-DET		
Baseline	65.1	74.0	57.0	19.88
+ SSB	65.8	74.2	42.9	15.02
+ BIM	66.0	74.4	51.1	18.93
+ D^2MR	66.4	74.9	58.6	21.92
+ SSB + BIM	66.8	75.1	**37.3**	**13.65**
+ SSB + D^2MR	67.1	75.7	44.7	17.64
+ BIM + D^2MR	67.5	75.9	58.3	21.69
+ SSB + BIM + D^2MR (w/o STN)	67.5	76.0	39.1	15.31
FastFlow-DETR	**68.0**	**76.4**	38.7	14.03

are summarized in Table III, where different module combinations are evaluated in terms of detection accuracy, computational complexity, and model size.

Starting from the baseline model, introducing SSB leads to a clear reduction in computational cost while slightly improving detection accuracy on both datasets. This indicates that selective spatial computation helps remove redundant operations and improves feature efficiency. Further incorporating BIM consistently enhances performance, especially when combined with SSB, where mAP increases to 66.8% on GC10-DET and 75.1% on NEU-DET, accompanied by a notable reduction in GFLOPs. This suggests that interweaving semantic and detail features during fusion is beneficial for detecting small and low-contrast defects.

When the D^2MR module is added, detection accuracy continues to improve across both datasets. The full combination of SSB, BIM, and D^2MR achieves 68.0% mAP on GC10-DET and 76.4% on NEU-DET, demonstrating the effectiveness of the proposed relay-based multi-scale encoding strategy in preserving fine-grained defect information. To further analyze the role of feature normalization, STN is removed from the full model configuration. Without STN, mAP drops consistently on both datasets while GFLOPs and parameter count remain nearly unchanged. This confirms that STN contributes to stabilizing feature amplitudes and preserving weak defect responses under complex background conditions. For a qualitative analysis, Fig. 8 (a) shows the mAP@50 improvements on GC10-DET and NEU-DET, and Fig. 8 (b) illustrates the corresponding changes in GFLOPs and parameter counts.

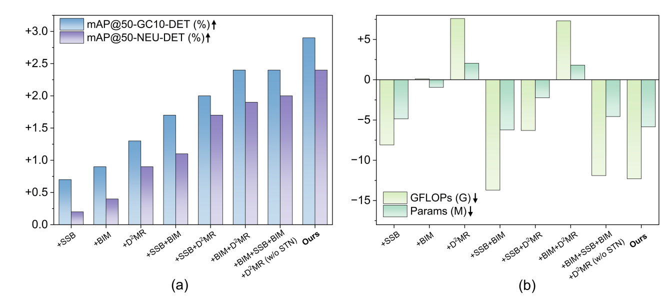

Fig. 8. Ablation results relative to the baseline on GC10-DET and NEU-DET. The vertical axis is the change relative to the baseline.

V. Conclusion

In this paper, we propose FastFlow-DETR, an efficient and lightweight framework for steel surface defect detection. It addresses challenges like multi-scale variations, weak defect signals, and high computational costs in industrial environments. To improve accuracy and efficiency, we restructure the RT-DETR architecture with key innovations. The SSB reduces redundant computations while enhancing defect-related features. The BIM improves feature fusion, aligning shallow detail features with deep semantic representations. The D^2MR module refines multi-scale interactions, preserving small-scale defect features. The STN stabilizes feature modulation, preventing weak defect signals from being over-compressed.

Experimental results on GC10-DET and NEU-DET show that FastFlow-DETR outperforms existing models, achieving superior mAP and high inference speed. Overall, it strikes an optimal balance between accuracy and computational efficiency, offering a practical solution for real-time, industrial-grade steel surface defect detection.

Acknowledgment

This study was supported by the Nature Science Foundation of China (62006210, 62206252), the Key Project of Public Benefit in Henan Province of China (201300210500), and the Key RD Program of Henan (251111212100).

References

[1] X. Wen, J. Shan, Y. He, and K. Song, "Steel surface defect recognition: a survey. coat 2079–6412 (13): 17," 2023.

[2] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[3] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[4] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

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

[6] M. Yaseen, "What is yolov8: An in-depth exploration of the internal features of the next-generation object detector," 2024. [Online]. Available: https://arxiv.org/abs/2408.15857

[7] C.-Y. Wang, I.-H. Yeh, and H.-Y. Mark Liao, "Yolov9: Learning what you want to learn using programmable gradient information," in *European conference on computer vision*. Springer, 2024, pp. 1–21.

[8] A. Wang, H. Chen, L. Liu, K. Chen, Z. Lin, J. Han, and G. Ding, "Yolov10: Real-time end-to-end object detection," 2024. [Online]. Available: https://arxiv.org/abs/2405.14458

[9] R. Khanam and M. Hussain, "Yolov11: An overview of the key architectural enhancements," 2024. [Online]. Available: https://arxiv.org/abs/2410.17725

[10] Y. Tian, Q. Ye, and D. Doermann, "Yolov12: Attention-centric real-time object detectors," *arXiv preprint arXiv:2502.12524*, 2025.

[11] J. Yu, X. Cheng, and Q. Li, "Surface defect detection of steel strips based on anchor-free network with channel attention and bidirectional feature fusion," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–10, 2021.

[12] Y. Du, H. Chen, Y. Fu, J. Zhu, and H. Zeng, "Aff-net: A strip steel surface defect detection network via adaptive focusing features," *IEEE Transactions on Instrumentation and Measurement*, vol. 73, pp. 1–14, 2024.

[13] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*. Springer, 2020, pp. 213–229.

[14] Y. Zhao, W. Lv, S. Xu, J. Wei, G. Wang, Q. Dang, Y. Liu, and J. Chen, "Detrs beat yolos on real-time object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2024, pp. 16 965–16 974.

[15] D.-c. Choi, Y.-J. Jeon, S. H. Kim, S. Moon, J. P. Yun, and S. W. Kim, "Detection of pinholes in steel slabs using gabor filter combination and morphological features," *Isij International*, vol. 57, no. 6, pp. 1045–1053, 2017.

[16] R. Manish, A. Venkatesh, and S. D. Ashok, "Machine vision based image processing techniques for surface finish and defect inspection in a grinding process," *Materials Today: Proceedings*, vol. 5, no. 5, pp. 12 792–12 802, 2018.

[17] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[18] W. Shi, C. Li, J. Dai, and N. Niu, "Gcf-net: Steel surface defect detection network based on global attention perception and cross-layer interactive fusion," *Electronics*, vol. 14, no. 9, p. 1776, 2025.

[19] H. Song, "Rstd-yolov7: a steel surface defect detection based on improved yolov7," *Scientific Reports*, vol. 15, no. 1, p. 19649, 2025.

[20] Z. Yang and Y. Liu, "A steel surface defect detection method based on improved retinanet," *Scientific Reports*, vol. 15, no. 1, p. 6045, 2025.

[21] X. Chen, X. Zhang, Y. Shi, and J. Pang, "Hct-det: A high-accuracy end-to-end model for steel defect detection based on hierarchical cnn–transformer features," *Sensors (Basel, Switzerland)*, vol. 25, no. 5, p. 1333, 2025.

[22] Y. Zhou and Z. Zhao, "Mpa-yolo: Steel surface defect detection based on improved yolov8 framework," *Pattern Recognition*, p. 111897, 2025.

[23] M. Lu, W. Sheng, Y. Zou, Y. Chen, and Z. Chen, "Wss-yolo: An improved industrial defect detection network for steel surface defects," *Measurement*, vol. 236, p. 115060, 2024.

[24] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable DETR: deformable transformers for end-to-end object detection," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. [Online]. Available: https://openreview.net/forum?id=gZ9hCDWe6ke

[25] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[26] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[27] X. Lv, F. Duan, J.-j. Jiang, X. Fu, and L. Gan, "Deep metallic surface defect detection: The new benchmark and detection network," *Sensors*, vol. 20, no. 6, p. 1562, 2020.

[28] K. Song and Y. Yan, "A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects," *Applied Surface Science*, vol. 285, pp. 858–864, 2013.

[29] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," *arXiv preprint arXiv:1904.07850*, 2019.

[30] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: Fully convolutional one-stage object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[31] S. Zhang, C. Chi, Y. Yao, Z. Lei, and S. Z. Li, "Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[32] R. Khanam and M. Hussain, "What is yolov5: A deep look into the internal features of the popular object detector," 2024. [Online]. Available: https://arxiv.org/abs/2407.20892