International Journal of Engineering Inventions
e-ISSN: 2278-7461, p-ISSN: 2319-6491
Volume 15, Issue 9 [September 2026] PP: 80-85

A Multiscale Feature Detection Method for Steel Surface Defects in Intelligent Manufacturing

Junhan Yang, Yuxin Liu, Wanyun Xie, Xinni Jiang, Qingzhi Yang

ABSTRACT: *To address the large scale variation, weak edge textures, and limited deployment resources associated with steel surface defects, this study develops an Adaptive Shape-Focusing Model (ASFM) based on YOLO11s. ADown, FDPN, DySample, PSConv, and MANet_FasterCGLU are introduced into the backbone and neck to improve detail retention during downsampling, multiscale feature interaction, and representation of edge regions. Experiments were conducted on the NEU-DET dataset. The original 1,800 images were expanded to 5,400 through data augmentation and divided into training and validation sets at a ratio of 8:2. Under identical training conditions, ASFM achieved an mAP of 97.3%, a Precision of 95.1%, a Recall of 92.8%, and an mAP@0.5:0.95 of 77.3%. These results exceeded the YOLO11s baseline by 5.1, 5.9, 8.1, and 9.1 percentage points, respectively, while the parameter count decreased from 9.4 M to 8.3 M. A visual detection system developed around the model supports single images, image batches, videos, and camera streams, providing an implementation basis for offline steel inspection and online edge-side detection.*

Keywords: *Steel surface defects; Object detection; Multiscale feature fusion; YOLO11s; Lightweight deployment*

Date of Submission: 02-09-2026 Date of acceptance: 12-09-2026

I. INTRODUCTION

Steel surface quality directly affects subsequent forming, coating, and service reliability. Hot-rolling and cold-rolling lines operate at high speeds, while defects such as cracks, scratches, pitted surfaces, patches, inclusions, and rolled-in scale exhibit wide scale variation, substantial intra-class diversity, and similar inter-class textures. Manual visual inspection is easily affected by fatigue, illumination, and production-line pace. Traditional threshold segmentation and handcrafted texture descriptors also struggle to adapt reliably to reflections, dust, vibration, and other industrial interference. Existing reviews identify small targets, limited samples, class imbalance, and real-time requirements as the principal challenges in steel surface defect recognition [1].

Convolutional neural networks have shifted object detection from handcrafted feature design to end-to-end feature learning. R-CNN and Faster R-CNN achieve strong localization through region proposals [2-3], but their two-stage pipelines introduce substantial inference overhead in high-throughput applications. YOLO integrates bounding-box regression and class prediction into a single network and thus provides a more direct route to real-time detection [4], while SSD also follows a one-stage multiscale prediction strategy [5]. In industrial defect images, shallow features preserve location and texture details, whereas deep features contain stronger semantics. The organization of these hierarchical features under a limited computational budget therefore determines whether small defects can be detected reliably. FPN constructs a semantically consistent feature pyramid through a top-down path and lateral connections [6], and BiFPN further improves efficiency through weighted bidirectional fusion [7]. However, because steel defects often have weak boundaries and skewed scale distributions, generic fusion structures may still allow local defect responses to be overwhelmed by the background.

The NEU surface defect database contains six categories of hot-rolled steel-strip defects, with 300 grayscale images of 200 x 200 pixels per category and object annotations for detection [8]. Studies of hierarchical feature fusion on this dataset show that combining location details with high-level semantics benefits both defect classification and localization [9]. Following this idea, the present study uses YOLO11s as the baseline and develops the ASFM multiscale feature detection model. The design focuses on information loss during downsampling, insufficient cross-scale interaction, and weak responses around defect edges. Comparative and ablation experiments are then conducted under consistent training conditions, and the detector is integrated into a desktop visualization system to establish a complete workflow from model training to result presentation.

The resulting research problem is therefore not merely to increase detector depth or width, but to preserve information that is easily lost when weak defects are repeatedly downsampled. A practical detector must retain fine spatial evidence, combine it with stable semantic cues, and avoid an excessive parameter

footprint. These requirements are closely connected: emphasizing only high-resolution features may increase computation and background sensitivity, whereas relying mainly on deep semantic features may suppress narrow cracks and low-contrast boundaries. The proposed design addresses this balance by assigning different modules to specific stages of the feature path. This stage-oriented organization also makes the contribution easier to evaluate because changes in downsampling, cross-scale fusion, upsampling, and channel selection can be examined through the stepwise ablation results rather than being treated as an indivisible network modification.

II. ASFM MULTISCALE FEATURE DETECTION METHOD

2.1 Overall Architecture

ASFM adopts the backbone, neck, and detection head of YOLO11s as its basic framework. Input images are resized to 640 x 640 pixels, normalized, and augmented by random flipping, rotation, and color perturbation before entering the backbone. The network produces feature maps at three scales, P3, P4, and P5, which encode high-resolution details, medium-scale morphology, and high-level semantics, respectively. FDPN focuses and diffuses features across levels in the neck, while DySample restores spatial resolution. The fused features are passed to the detection head to predict class probabilities and bounding-box coordinates. Non-maximum suppression then produces the final defect categories, locations, and confidence scores.

The architecture is organized around a clear division of responsibilities. The backbone extracts progressively abstract representations while ADown and PSConv reduce the loss of small-scale texture during resolution changes. The neck then becomes the principal location for exchanging information among P3, P4, and P5, because these feature maps contain complementary spatial and semantic content. FDPN controls this exchange, and DySample reconstructs higher-resolution representations without relying on a fixed interpolation pattern. Before prediction, MANet_FasterCGLU filters the fused channels so that the detection head receives a compact representation dominated by defect-relevant responses. This ordering is important because an enhancement applied after information has already disappeared cannot fully recover the original texture. ASFM therefore places preservation and interaction mechanisms before the final prediction stage rather than using the detection head alone to compensate for earlier losses.

Table 1. Main Components and Functions of ASFM

Module	Position	Main Function
ADown	Backbone	Multi-path pooling-convolution downsampling that reduces parameters while preserving contextual information
PSConv	Backbone	Asymmetric and dilated convolution that enlarges the receptive field and enhances fine-grained textures
FDPN	Neck	Focuses and diffuses key hierarchical features to improve multiscale interaction
DySample	Upsampling layer	Learns sampling offsets to reduce edge-detail loss caused by fixed interpolation
MANet_FasterCGLU	Before detection head	Performs multiscale aggregation and gated selection to strengthen weak-edge responses

2.2 Feature Extraction and Lightweight Downsampling

Successive convolution and pooling enlarge the receptive field but may gradually suppress cracks and pitted defects that occupy only a few pixels. ADown uses multi-path downsampling composed of average pooling, max pooling, and convolution. Average pooling retains regional statistics, max pooling emphasizes strong local responses, and the convolution branch performs channel transformation. The branch outputs are fused along the channel dimension, allowing the network to preserve texture discontinuities and local contrast while reducing spatial resolution. The structure also contributes to model lightweighting, reducing the parameter count from 9.4 M in the baseline to 8.3 M in the complete model.

PSConv strengthens fine-grained feature extraction during downsampling. Conventional square kernels cover elongated directional defects inefficiently, whereas asymmetric convolutions can emphasize horizontal and vertical textures separately. Dilated sampling further expands the effective receptive field, introducing broader context without an excessive increase in parameters. For cracks and scratches only a few pixels wide, this larger receptive field helps distinguish continuous defect textures from random noise. For isolated small targets such as pitted surfaces, the high-resolution branch retains local grayscale variation.

2.3 Multiscale Focusing and Dynamic Upsampling

FDPN enables bidirectional interaction among P3, P4, and P5. During feature focusing, hierarchical information is redistributed according to the response at the current scale. High-resolution features receive semantic constraints, while low-resolution features are supplemented with edge and location details. During feature diffusion, the focused representations are propagated to adjacent scales, reducing the localization of information caused by fusion along only one path. Compared with direct addition, this process places greater

emphasis on valid responses within defect regions and mitigates interference from strong reflections or background textures at the detection head.

During spatial-resolution recovery, fixed nearest-neighbor or bilinear interpolation cannot adjust sampling locations according to image content. DySample learns offsets from a point-sampling perspective, has low computational overhead, and has demonstrated general applicability in multiple dense prediction tasks [10]. ASFM applies DySample in the neck so that sampling points shift adaptively toward defect contours and texture discontinuities. MANet_FasterCGLU then performs multiscale aggregation and gated selection on the fused features, suppressing redundant channels while retaining information relevant to defect classification and boundary regression. The modules are not simply stacked. Instead, they operate on downsampling, cross-scale transfer, resolution recovery, and pre-output selection, forming a continuous information-preservation chain.

Together, these operations establish a controlled information flow from local texture preservation to global semantic interpretation. ADown first limits destructive compression, PSConv expands directional context, FDPN coordinates features at adjacent scales, and DySample aligns the recovered resolution with image-dependent structures. The gated aggregation stage then reduces redundancy before classification and regression. This sequence is particularly relevant to steel surfaces because a defect may occupy only a narrow band while the surrounding material produces strong repetitive texture. If early layers remove the narrow response, later fusion cannot reconstruct it reliably; if fusion preserves every response without selection, background patterns may reach the prediction head with comparable strength. The coordinated design therefore seeks to retain weak but consistent defect cues while suppressing responses that are spatially or semantically unsupported.

III. EXPERIMENTAL DESIGN AND RESULTS

3.1 Dataset and Experimental Setup

Experiments were conducted on the NEU-DET dataset, which contains six defect categories: rolled-in scale, cracks, pitted surfaces, patches, scratches, and inclusions. The original dataset contains 1,800 images. Random flipping, rotation, and noise augmentation expanded the dataset to 5,400 images, including 4,320 training images and 1,080 validation images. All models used an input size of 640 x 640, 300 training epochs, and a batch size of 8. Stochastic gradient descent was used for optimization, and the learning rate was adjusted with cosine annealing. The experimental environment consisted of Windows 11, PyTorch 1.12.1, CUDA 11.3, and an NVIDIA GTX 1650 GPU.

The evaluation metrics included Precision, Recall, mAP, mAP@0.5:0.95, parameter count, and GFLOPs. Precision represents the proportion of predicted defects that are detected correctly, whereas Recall measures the proportion of ground-truth defects that are identified. mAP summarizes the precision-recall curves across categories, and mAP@0.5:0.95 averages performance over multiple intersection-over-union thresholds, imposing stricter requirements on localization quality. Parameter count measures model storage requirements, while GFLOPs estimates the computational load of a single forward inference pass.

A fair comparison also requires the training and evaluation procedures to remain consistent across the baseline and modified networks. The same image size, dataset split, training duration, batch size, optimizer, and learning-rate schedule were therefore used for all reported models. Under this arrangement, performance differences are more reasonably associated with architectural changes than with unequal training budgets. The selected metrics describe complementary aspects of performance. Precision and Recall reveal the balance between false alarms and missed defects, while mAP summarizes category-level detection quality. The stricter mAP@0.5:0.95 measure is sensitive to localization across a range of overlap thresholds. Parameter count and GFLOPs extend the comparison beyond accuracy and provide evidence about storage demand and computational workload, which are essential when the intended application includes edge-side deployment.

3.2 Comparison with the Baseline Model

Table 2. Performance Comparison between YOLO11s and ASFM

Model	mAP	Precision	Recall	mAP@0.5:0.95	Parameters (M)	GFLOPs
YOLO11s	92.2%	89.2%	84.7%	68.2%	9.4	21.3
ASFM	97.3%	95.1%	92.8%	77.3%	8.3	22.4
Change	+5.1	+5.9	+8.1	+9.1	-11.7%	+5.2%

Table 2 shows that ASFM achieved an mAP of 97.3%, 5.1 percentage points higher than YOLO11s. Precision and Recall increased by 5.9 and 8.1 percentage points, respectively. The larger gain in Recall indicates that preserving multiscale details has a more direct effect on reducing missed detections. mAP@0.5:0.95 increased by 9.1 percentage points, which exceeded the gain in mAP at a single threshold. This result indicates that the improved model not only identifies defect categories more accurately but also improves the overlap between predicted bounding boxes and ground-truth targets.

The parameter count decreased by 11.7%, whereas GFLOPs increased from 21.3 to 22.4. The opposite trends reflect a trade-off between lightweight parameter design and the computational cost of feature fusion. ADown and the gating structure compress learnable parameters, while FDPN and dynamic upsampling introduce additional operations. This trade-off is feasible for edge devices with limited memory and storage but moderate computing capability. For higher-speed online production lines, pruning, quantization, or inference-engine optimization should be combined with the model to reduce latency further.

From a deployment perspective, parameter count and GFLOPs describe different resource constraints and should not be interpreted as interchangeable indicators. A smaller parameter set reduces model storage and weight-transfer requirements and can lower memory pressure during loading. GFLOPs, in contrast, approximates the arithmetic work performed during inference and is influenced by the repeated feature-fusion and sampling operations introduced in the neck. The results in Table 2 therefore indicate a model that is lighter in stored parameters but slightly more demanding computationally. Whether this trade-off is acceptable depends on the target device and production rhythm. A device with limited memory but adequate parallel computing capability may benefit from the design, whereas a latency-critical line may require operator fusion, reduced-precision inference, structured pruning, or hardware-specific compilation before deployment.

3.3 Ablation Results

Table 3. Stepwise Ablation Results of the Main Components

Model configuration	FDPN	DySample	PSConv	MANet	ADown	mAP
YOLO11s baseline	-	-	-	-	-	92.2%
Configuration 1	Yes	-	-	-	-	96.4%
Configuration 2	Yes	Yes	-	-	-	96.8%
Configuration 3	Yes	Yes	Yes	-	-	97.3%
Configuration 4	Yes	Yes	Yes	Yes	-	97.4%
ASFM	Yes	Yes	Yes	Yes	Yes	97.3%

In the stepwise ablation study, adding FDPN increased mAP from 92.2% to 96.4%, indicating that multiscale feature organization was the main source of performance improvement. DySample and PSConv further increased mAP to 97.3%, consistent with their design goals of detail recovery and receptive-field expansion. MANet briefly raised the metric to 97.4%. After ADown was added, mAP was 97.3%, a change of only 0.1 percentage points, while the parameter count of the complete model decreased to 8.3 M. ADown therefore contributes primarily to model-size control rather than an isolated accuracy gain. The results also show that an ablation analysis should not compare only the highest mAP; parameter count, computational cost, and deployment conditions must be considered together.

IV. DETECTION SYSTEM IMPLEMENTATION

To evaluate engineering usability, a desktop detection system was developed using Python, Ultralytics, and PyQt5. At startup, the program loads the trained weight file and uses a unified inference interface to process single images, image folders, video files, and camera streams. Users can adjust the confidence and intersection-over-union thresholds and select either a CPU or GPU. The interface displays predicted classes, bounding boxes, confidence scores, and target counts. The official YOLO11s implementation supports training, validation, inference, and model export, providing a unified software interface for subsequent deployment [11].

The system workflow consists of input management, model inference, result visualization, and data storage. Input management reads images from paths containing Chinese characters, parses videos frame by frame, and connects to cameras. The inference module performs preprocessing, forward computation, and non-maximum suppression. The visualization module draws class-specific detection boxes and lists locations and confidence scores in a table. The storage module exports annotated images or videos. Batch mode saves the result for each image and supports forward and backward browsing, while video mode preserves the frame sequence so that false and missed detections in continuous footage can be reviewed.

The current implementation is suitable for offline laboratory sampling inspection and model validation. Deployment on a production line would require industrial-camera triggering, lighting control, PLC coordination, and detection logs to be synchronized, followed by remeasurement of end-to-end latency on the target hardware. Model forward time should be distinguished from the time consumed by image acquisition, decoding, rendering, and storage so that a single model metric is not mistaken for overall system throughput. Long-term operation would also require fault recovery, weight-version management, and quality-traceability interfaces so that detection results can be transferred to a manufacturing execution system (MES) or quality management system (QMS).

A production-oriented implementation also requires the software pipeline to maintain consistency between model outputs and quality-control records. Each detected object should remain associated with its

source image or frame, category, confidence score, and bounding-box coordinates so that the decision can be reviewed later. For video or camera input, frame ordering and acquisition timestamps are necessary to connect a defect event with the corresponding strip position and production status. Threshold configuration should be stored with the result because changing confidence or overlap thresholds can alter both the number of alarms and the apparent defect distribution. These requirements do not change the detector itself, but they determine whether its outputs can support traceability and process improvement. The desktop system provides a functional verification platform, while production deployment would require reliable interfaces, persistent logging, controlled model versions, and fault-handling mechanisms around the inference core.

V. DISCUSSION

The experimental results support the effectiveness of multiscale feature focusing for detecting small steel surface defects, but three limitations remain. First, training and validation were performed mainly on a public dataset. Its image dimensions, illumination, and background distribution do not fully represent actual production lines, and data augmentation can only approximate reflections and noise rather than replace on-site sampling. Second, the comparison focuses on offline accuracy, parameter count, and computational complexity; inference latency, GPU memory usage, and energy consumption have not yet been tested on standardized hardware. Third, the stepwise ablation study demonstrates changes among component combinations but cannot fully separate interactions between modules. Future work may use a full-factorial design or pairwise combinations of key modules to determine whether accuracy gains depend on specific structures.

Steel defect detection also faces ambiguous class boundaries and distribution shifts. Scratches within the same category may appear as horizontal, vertical, or oblique textures, while different defect categories may exhibit similar grayscale appearances [8]. Further research should therefore proceed from both the data and model perspectives. On the data side, an on-site dataset should cover variations in material, line speed, illumination, and equipment while retaining difficult cases and false detections. On the model side, knowledge distillation, structured pruning, and quantization-aware training can be incorporated while preserving the ASFM multiscale structure, thereby producing model variants for different edge devices. Detection thresholds should also be set according to product grade and defect cost rather than inherited from fixed defaults.

These limitations suggest that future evaluation should be organized as a staged transition from public-dataset validation to production-line verification. The first stage should reproduce the reported metrics under controlled conditions and confirm that model conversion does not change prediction behavior. The second stage should introduce samples collected under different illumination, speeds, materials, and surface finishes to measure sensitivity to domain shift. The final stage should evaluate end-to-end throughput, alarm stability, and traceability under continuous operation. Error analysis should separate missed detections, incorrect classifications, and inaccurate localization because each failure type points to a different corrective action. Missed weak defects may require improved sampling or training data, class confusion may require better category representation, and unstable boxes may require localization-oriented optimization. This staged procedure would connect model-level accuracy with the operational reliability expected from an industrial inspection system.

VI. CONCLUSION

This study developed a YOLO11s-based ASFM multiscale feature detection model for small steel surface defects, weak-edge defects, and resource-constrained deployment. ADown, PSConv, FDPN, DySample, and MANet_FasterCGLU improve detail retention during downsampling, receptive-field coverage, multiscale fusion, upsampling, and gated feature selection, respectively. On NEU-DET, the model achieved an mAP of 97.3%, 5.1 percentage points higher than the baseline, while reducing the parameter count by 11.7%. The accompanying visualization system supports single-image, batch-image, video, and camera-based detection. The present conclusions are limited to the specified dataset and experimental environment. Future work should evaluate robustness and end-to-end latency using data collected from actual production lines and should improve lightweight deployment and quality-traceability functions.

REFERENCES

[1] WEN X, SHAN J, HE Y, et al. Steel Surface Defect Recognition: A Survey[J]. Coatings, 2023, 13(1): 17.
[2] GIRSHICK R, DONAHUE J, DARRELL T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE, 2014: 580-587.
[3] REN S, HE K, GIRSHICK R, et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, 39(6): 1137-1149.
[4] REDMON J, DIVVALA S, GIRSHICK R, et al. You Only Look Once: Unified, Real-Time Object Detection[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE, 2016: 779-788.
[5] LIU W, ANGUELOV D, ERHAN D, et al. SSD: Single Shot MultiBox Detector[C]//Computer Vision - ECCV 2016. Cham: Springer, 2016: 21-37.
[6] LIN T Y, DOLLAR P, GIRSHICK R, et al. Feature Pyramid Networks for Object Detection[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE, 2017: 2117-2125.

[7] TAN M, PANG R, LE Q V. EfficientDet: Scalable and Efficient Object Detection[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE, 2020: 10781-10790.
[8] SONG K, YAN Y. A Noise Robust Method Based on Completed Local Binary Patterns for Hot-Rolled Steel Strip Surface Defects[J]. Applied Surface Science, 2013, 285: 858-864.
[9] 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, 2020, 69(4): 1493-1504.
[10] LIU W, LU H, FU H, et al. Learning to Upsample by Learning to Sample[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. Piscataway: IEEE, 2023: 6004-6014.
[11] JOCHER G, QIU J. Ultralytics YOLO11[CP/OL]. (2024-09-10)[2026-09-07]. https://github.com/ultralytics/ultralytics.