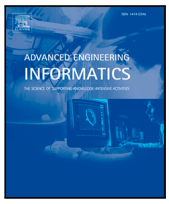
Advanced Engineering Informatics 69 (2026) 103891

Contents lists available at ScienceDirect

Advanced Engineering Informatics

journal homepage: www.elsevier.com/locate/aei



Full length article

An accurate and resource-efficient network for surface anomaly detection via enhanced downsampling and activation representation

Xunkuai Zhou [a,b], Xi Chen [b,*], Jie Chen [a,c], Ben M. Chen [b]

[a] School of Electronics and Information Engineering, Tongji University, Jiading, Shanghai 201804, China
[b] Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong, China
[c] Harbin Institute of Technology, Harbin, 150001, China

ARTICLE INFO

Keywords:
Anomaly detection
Downsample enhancement
Activation function
Edge computing
Contextual feature fusion

ABSTRACT

In deep learning-based anomaly detection, performance degradation often results from feature loss caused by downsampling and the limited nonlinear representation capacity of conventional activation functions. Moreover, improving detection accuracy often demands substantial computational resources, thereby hindering practical deployment in real-time and resource-constrained collaborative multi-robot inspection settings. To address these challenges, this paper proposes DEANet, a memory-efficient and real-time anomaly detection methodology with four key components, designed to achieve high accuracy at low computational cost. First, a lightweight feature aggregation neck improves feature fusion efficiency while reducing computational overhead. Second, a contextual feature extraction module leverages environmental semantics to enhance both detection and localization accuracy. Third, to alleviate the feature degradation introduced by hierarchical downsampling, two enhancement modules are designed to facilitate a better trade-off between accuracy and computational cost. Fourth, we propose a parameterized activation function (ACLU) that enhances the network's nonlinear representational capacity. ACLU achieves higher accuracy and demonstrates faster convergence compared to recent advanced activation functions. Experiments on three benchmark datasets confirm that DEANet achieves state-of-the-art accuracy with only 2.8 million parameters, while reducing training data demands by 70%, parameter count by 87.8%, and computational cost by 92.6%, demonstrating its strong efficiency–performance trade-off under resource-constrained conditions. Edge-computing deployment tests validate DEANet's real-time performance at 52.1 FPS. These results highlight DEANet's practicality and scalability for deployment in real-world, resource-constrained settings. The source code will be available at https://github.com/chriszxk/surface-detection.git.

1. Introduction

Surface anomaly detection or fault diagnosis plays a vital role in construction and industrial products by enhancing lifecycle management, facilitating timely maintenance, and ensuring quality assurance [1,2]. In recent years, deep learning-based methods for anomaly detection have advanced rapidly. Multi-robot collaboration system equipped with these techniques has improved the efficiency of surface anomaly detection in large-scale buildings and industrial facilities, while also enhancing operational safety. However, achieving high detection accuracy under constrained computational resources remains a major challenge. This limitation highlights the need for anomaly detection methods that are both accurate and computationally efficient, thereby extending the operational lifespan of resource-constrained systems. Consequently, efficient and reliable surface anomaly detection has emerged as a critical research priority for collaborative multi-robot inspection systems.

Despite significant progress in deep learning-based anomaly detection, several challenges persist. First, most detection methods reduce computational costs (computational complexity) by performing feature downsampling through convolutional or pooling layers [3,4]. For fine, elongated architectural cracks with minimal pixel occupancy, such downsampling methods often lead to a degradation in detection accuracy due to feature loss or insufficient representation capacity. While increasing the parameter count (space complexity) can enhance feature representation capability, it also leads to higher memory consumption and increased computational costs. Therefore, achieving an effective trade-off between computational cost, parameter count, and detection accuracy remains a critical challenge for anomaly detection

* Corresponding author.
E-mail addresses: xunkuaizhou@cuhk.edu.hk, 2010474@tongji.edu.cn (X. Zhou), xichen002@mae.cuhk.edu.hk (X. Chen), chenjie206@tongji.edu.cn (J. Chen), bmchen@cuhk.edu.hk (B.M. Chen).

https://doi.org/10.1016/j.aei.2025.103891
Received 23 May 2025; Received in revised form 27 August 2025; Accepted 12 September 2025

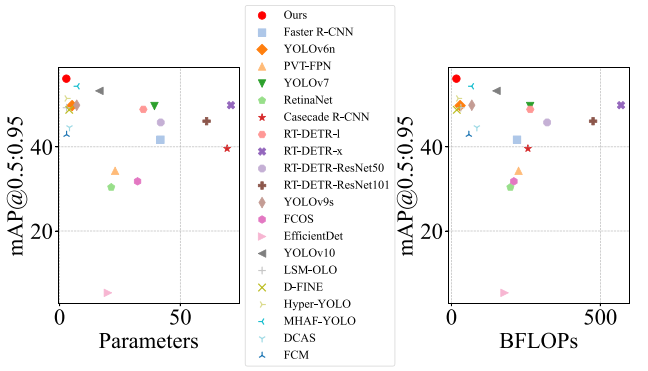


Fig. 1. Left: accuracy vs. parameter count; Right: accuracy vs. computational cost. The proposed method (red dot) achieves a favorable trade-off between low complexity and favorable performance, making it suitable for deployment on resource-constrained platforms, especially in multi-agent settings.

networks, particularly when deployed on edge-computing devices in robotic systems.

Second, contextual environmental semantics are often overlooked in existing approaches [5]. As certain damages tend to occur under specific environmental conditions, incorporating contextual environmental information can significantly enhance both anomaly detection accuracy and localization performance, particularly in few-shot anomaly detection scenarios. Leveraging such contextual cues contributes to more robust and accurate anomaly detection.

Third, lightweight detection methods often lack effective information fusion mechanisms, which can lead to uneven feature representation, particularly for slender crack defects. Effectively integrating the information extracted by the backbone network is a critical step in enhancing feature visibility and, consequently, improving detection performance. Employing deeper neural networks for feature fusion significantly increases the parameter count (space complexity) and imposes a heavy burden on storage and computation in resource-constrained platforms [6]. Designing an effective lightweight feature fusion network can alleviate storage capacity constraints while improving detection performance.

Furthermore, our evaluation of several advanced activation functions indicates that their representational capacity is often limited, in some cases even leading to degraded detection performance [7–9]. However, certain activation functions have shown effectiveness in specific neural networks; our experiments identified instability of activation failure [10], which compromises the representational capacity for complex anomaly data and consequently reduces detection accuracy. Therefore, further improvements in the design of activation functions are necessary.

To address these challenges, this work proposes a surface anomaly detection framework, termed the Downsampling-Enhanced and Activation Representation Network (DEANet), which aims to achieve a balance between computational cost (computational complexity), parameter efficiency (space complexity) and detection accuracy, thereby enabling deployment on edge-computing platforms. To achieve accurate and real-time anomaly detection while maintaining memory efficiency, we utilize a contextual feature extraction module (CFM) that captures contextual semantics to enhance detection accuracy. A downsampling enhancement module (DEM) is also introduced to strengthen feature extraction capability, thereby improving detection performance.

Lastly, we propose a novel activation function that enhances the network's nonlinear representation capability without increasing memory overhead. To further reduce computational cost and parameter count, a lightweight feature aggregation network is employed for efficient feature fusion. In addition, a lightweight downsampling module (LDM) is integrated alongside the downsampling enhancement module (DEM) to maintain an optimal trade-off between computational cost and detection accuracy. As illustrated in Fig. 1, DEANet demonstrates clear advantages in the trade-off between detection accuracy and parameter count (left) and computational cost (right). In the figure, the vertical axis represents detection accuracy. In contrast, the horizontal axis corresponds to the number of parameters and the computational cost (measured in billion floating-point operations per second, or BFLOPs), respectively. Notably, DEANet attains competitive accuracy compared to state-of-the-art networks while requiring significantly fewer training samples (See Section 4.2).

In summary, the main contributions of this paper are as follows:

(1) We propose an accurate and real-time anomaly detection network that incorporates environmental context information to enhance both localization and detection performance. Furthermore, a lightweight feature aggregation module is employed to improve the efficiency for feature fusion, enabling practical deployment on resource-constrained robotic edge computing devices.
(2) To alleviate the performance degradation caused by feature loss during hierarchical downsampling, we introduce two downsampling enhancement modules. Their combined integration facilitates a favorable balance between detection accuracy and computational cost, thereby supporting practical deployment in resource-constrained settings.
(3) We develop an activation function that adapts to the specific activation requirements of diverse neural network architectures. Experimental results demonstrate that it outperforms 20 state-of-the-art activation functions in terms of both accuracy and generalization capability, and achieves a faster convergence rate compared to the advanced activation function.
(4) Extensive qualitative and quantitative evaluations demonstrate that the proposed network outperforms existing methods in surface anomaly detection. Notably, it achieves a 70% reduction in required training samples while maintaining superior detection performance. Edge-computing deployment confirms real-time inference at 52.1 FPS, underscoring its practical applicability in industrial settings.

The remainder of this paper is organized as follows. Section 2 reviews related work and existing literature. Section 3 outlines the theoretical foundation and details the model development process. Section 4 presents the experimental setup, results, and performance evaluation. Section 5 discusses the limitations of our study and outlines future work. Finally, Section 6 summarizes the key findings and discusses the main conclusions.

2. Literature review

2.1. Deep learning-based anomaly detection

Anomaly detection in high-rise infrastructure and large-scale industrial facilities remains a critical challenge for maintaining structural reliability and product integrity [11]. Extensive research efforts have focused on addressing this issue using deep learning-based image processing techniques. For instance, Block et al. [12] proposed a framework for detecting and classifying anomalies on stamped metal surfaces by exploring temporal coherence in consecutive frames through tracking detected regions. Xie et al. [13] introduced FFCNN, a deep neural network designed for detecting surface anomaly in magnetic materials, effectively addressing both efficiency and cost concerns. Sun et al. [14] presented a deep learning-based approach for weld anomaly detection, although the method is limited by relatively slow inference speed. Attention mechanisms have also been incorporated to improve detection accuracy, as they selectively emphasize discriminative features and suppress irrelevant information, thereby strengthening feature representation. For example, Hu et al. [15] integrated an object-level attention module into their training strategy for accurate casting anomaly

detection. However, many of these approaches overlook a key limitation: the degradation of accuracy caused by aggressive downsampling operations. To address robustness, Zhou et al. [16] combined a visual attention model with wavelet transforms. Nevertheless, the method emphasizes detection accuracy at the expense of computational efficiency, which hampers its deployment on resource-constrained devices.

Furthermore, the majority of existing techniques focus predominantly on crack detection. In real-world applications, however, anomalies in infrastructure and industrial systems can manifest in more diverse forms, such as spalling, delamination, and moisture intrusion. Detecting a broad spectrum of anomalies increases spatial and computational complexity, necessitating generalizable and efficient frameworks to manage the challenges of real-world scenarios.

2.2. Feature fusion

Numerous techniques have been proposed to enhance automated anomaly detection, among which multi-scale feature fusion combined with attention mechanisms has proven particularly effective [17]. For example, fusion strategies based on forgotten inputs have demonstrated potential in mitigating class imbalance by reinforcing contextual semantics without degrading spatial resolution [18]. However, such methods often introduce considerable computational overhead, limiting their applicability in resource-constrained settings. BDDN [19] leverages ResNet-50 as a backbone and employs a dual-attention feature pyramid network for feature fusion; however, the model comprises over 25 million parameters, posing a significant challenge for deployment on memory-constrained devices. Similarly, DMF-Net [20] employs a dual-encoded multi-scale fusion strategy to enhance anomaly detection for variable shapes and sizes, thereby improving feature representation. Despite these strengths, the applicability of DMF-Net remains limited, as it primarily targets road surface cracks.

Moreover, most existing fusion approaches fail to incorporate the environmental context in which damages occur, despite certain damage types being strongly correlated with specific environmental conditions. Integrating environmental semantics into feature fusion strategies has the potential to improve both the localization and identification of diverse anomalies, especially in complex operational scenarios.

2.3. Activation function

Activation functions, as essential elements of neural networks, introduce nonlinearity and thereby facilitate the modeling of complex relationships. Among them, the Rectified Linear Unit (ReLU) is widely adopted due to its ability to improve inference efficiency and facilitate the training efficiency of deep neural networks. However, ReLU suffers from the issue of "dead neurons", wherein neurons become inactive and cease to update during training. To mitigate this, several ReLU variants have been proposed [21–26], which typically introduce a slight nonzero gradient for negative inputs or incorporate trainable parameters to adapt the slope. Despite these advances, gaps remain in the activation landscape that affect learning dynamics in anomaly detection networks.

TinyReLU [9], a function specifically designed for fine crack detection, introduces a modified derivative behavior tailored to the characteristics of such defects. However, our analysis of its derivative reveals that it approaches zero at both negative and positive infinity. According to the chain rule, this vanishing gradient effect impedes effective gradient backpropagation, particularly in deeper architectures, thereby limiting the activation range and network convergence. Experimental evaluations further confirm that TinyReLU can lead to training instability. These observations indicate the necessity for more generalizable activation functions that maintain gradient flow and support effective learning across diverse input distributions.

In addition to the limitations above, most existing studies have not systematically evaluated the few-shot learning capabilities of anomaly detection methods. This gap is particularly significant in real-world scenarios, where data availability is inherently limited and annotation processes are costly. Assessing detection performance under few-shot conditions offers critical insights into a model's generalization ability and robustness in data-scarce environments, which are essential in industrial and infrastructure monitoring applications.

3. Methodology

3.1. Overview of framework

As illustrated in Fig. 2, the proposed anomaly detection framework comprises two main components: feature extraction and feature fusion. In the feature extraction stage, three custom-designed modules are introduced: the Contextual Feature Extraction Module (CFM), the Lightweight Downsampling Module (LDM), and the Downsampling Enhancement Module (DEM). Two convolutional layers initially process the input image to extract low-level features, which are subsequently refined through the sequential application of these modules.

To enhance multi-scale object detection, a SPPF module is integrated to extract hierarchical features at multiple scales [27]. In parallel, the C2PSA module [28], incorporating both channel and spatial attention mechanisms, is employed to enrich the feature representation. These processed features are then passed through an aggregation network for fusion. Finally, the detection head outputs the object's confidence score, predicted class, and bounding box coordinates.

3.2. Contextual feature extraction module (CFM)

In remote sensing imagery, object classification can be challenging when distinct object categories share similar visual characteristics. For example, cars and ships often exhibit comparable shapes and colors, making them difficult to differentiate based solely on appearance. However, incorporating contextual information such as the surrounding environment can significantly enhance classification accuracy. A car typically appears on roads or in urban areas, while a ship is generally located in water regions.

Analogously, certain types of anomaly occur preferentially in specific environments. For instance, moisture-induced anomalies are commonly found in humid or water-exposed areas. Recognizing this association, we design the Contextual Feature Extraction Module (CFM) to incorporate environmental semantics into the feature representation process.

As shown in the central lower region of Fig. 2, CFM comprises two submodules: the Dual Convolutional Module (DCM) and DCM1. The DCM employs two convolutional branches with a dilation rate of 4 to expand the receptive field without increasing parameter complexity, as opposed to conventional large-kernel convolutions. DCM1 includes three convolutional layers and one DCM unit. Specifically, the input features are evenly divided into two parallel branches in DCM1. The first branch consists of a convolutional layer followed by a DCM unit, while the second branch includes a single convolutional layer. The outputs from both branches are concatenated and further processed by a convolutional layer to generate the final contextual feature representation. As illustrated in Fig. 3, our CFM effectively enlarges the receptive field (right), thereby enhancing the representation of fine-grained anomalies and complex backgrounds, improving the sufficiency of feature extraction, and ultimately boosting anomaly detection accuracy.

3.3. Downsampling enhancement module (DEM)

Downsampling operations in convolutional neural networks often lead to information loss, resulting in a degradation in detection accuracy and localization accuracy. To mitigate this issue, we propose the Downsampling Enhancement Module (DEM). As illustrated in Fig.

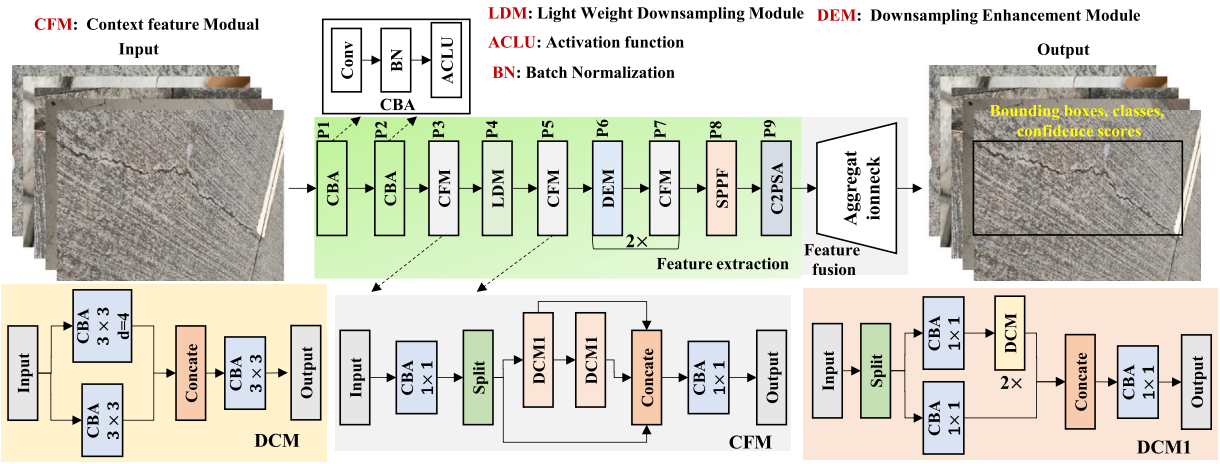


Fig. 2. Anomaly detection framework. This framework conducts feature extraction and enhances feature representation through the extractor. Subsequently, the Aggregation Neck facilitates enhanced feature fusion (CBA, Conv + BN + ACLU).

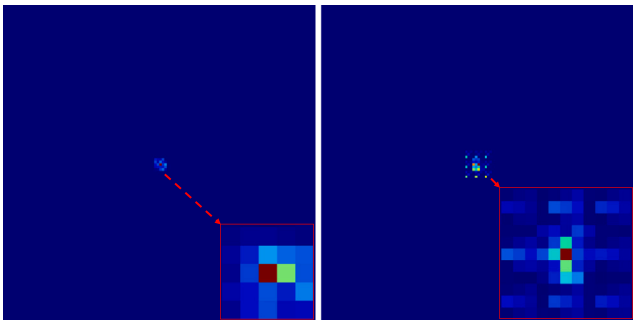


Fig. 3. The Effective Receptive Fields of without CFM (left) and with CFM (right). Our CFM achieves the largest field. This improves the extractor's ability to capture low frequencies.

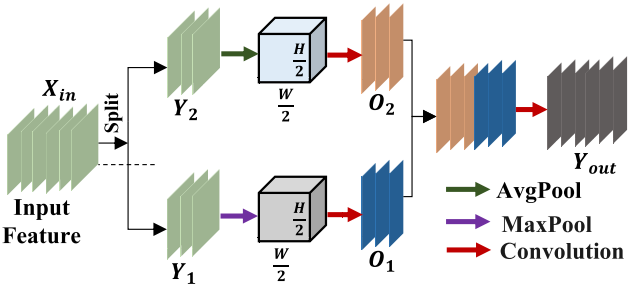


Fig. 4. The architecture of the DEM module.

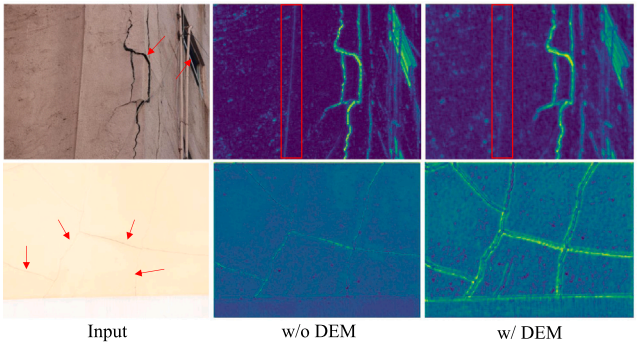


Fig. 5. The figure demonstrates that the DEM effectively mitigates feature degradation introduced by downsampling and improves the discriminative power of the extracted features. The arrow denotes the anomalous region.

4, the DEM architecture is designed to preserve both high- and low-frequency features during spatial resolution reduction.

Specifically, the input feature map $X_{\text{in}} \in \mathbb{R}^{W\times H\times C}$ is split along the channel dimension into two equal parts, denoted as Y_1 and Y_2, each with a size of $\mathbb{R}^{W\times H\times \frac{C}{2}}$. The bottom branch applies max-pooling and then a 1×1 convolution to Y_1, which enhances the capture of high-frequency components. The upper branch processes Y_2 using average pooling and a 3×3 convolution to emphasize low-frequency features. Motivated by the observation that small convolution kernels are more effective at extracting fine-grained, high-frequency details, while larger kernels are better suited for low-frequency patterns, we design each branch accordingly. The resulting features $O_1, O_2 \in \mathbb{R}^{0.5\ W\times 0.5H\times 0.5C}$ are concatenated and passed through a final 3×3 convolutional layer to generate the output feature map $Y_{\text{out}} \in \mathbb{R}^{0.5\ W\times 0.5H\times C_1}$, where C_1 denotes the output channel dimension.

The overall computation process of the DEM can be expressed as:

$$\begin{aligned} &Y_1, Y_2 = Split(X_{in}) \\ &O_1 = CBA_{1\times1}(MaxPool(Y_1)) \\ &O_2 = CBA_{3\times3}(AvgPool(Y_2)) \\ &Y_{out} = CBA_{3\times3}(Concat[O_1, O_2]) \end{aligned} \tag{1}$$

where $Y_1 \in \mathbb{R}^{W\times H\times 0.5C}$ and $Y_2 \in \mathbb{R}^{W\times H\times 0.5C}$ are the two parts separated from the input X_{in}. $O_1 \in \mathbb{R}^{0.5\ W\times 0.5H\times 0.5C}$ and $O_2 \in \mathbb{R}^{0.5\ W\times 0.5H\times 0.5C}$ are the outputs computed from the bottom and upper branches for Y_1 and Y_2, respectively. $Y_{\text{out}} \in \mathbb{R}^{0.5\ W\times 0.5H\times C1}$ represents the final output feature of the DEM. CBA denotes a sequential module comprising a Convolutional layer, followed by Batch Normalization, and the ACLU activation function (See Section 3.6).

Fig. 5 presents, from left to right, the input image, the feature map without DEM, and the feature map with DEM. Relative to the no-DEM case, the DEM-enabled features are more pronounced (second row) and exhibit improved representation quality (first row). In particular, in the first row, the model without DEM inadvertently highlights corner responses (marked by red box), whereas the model with DEM successfully suppresses these redundant features. The qualitative comparison indicates that DEM selectively enhances anomaly-relevant responses while suppressing spurious corner activations, thereby increasing the feature signal-to-noise ratio and improving representation quality and localization.

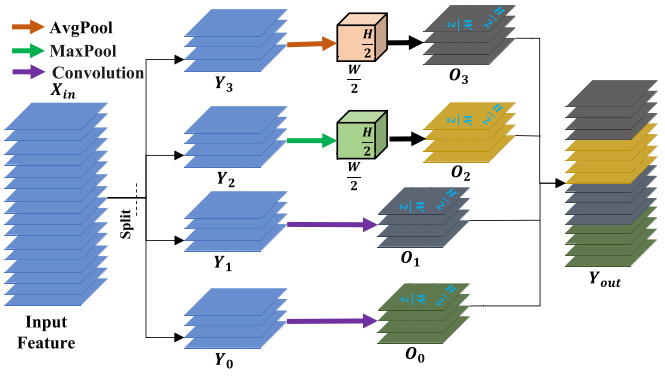


Fig. 6. The architecture of the LDM module.

3.4. Lightweight downsampling module (LDM)

To reduce the computational cost of the model while further enhancing detection performance, we design a Lightweight Downsampling Module (LDM) that operates in conjunction with the DEM to achieve a balance between computational cost and accuracy. As illustrated in Fig. 6, the LDM draws inspiration from multimodal feature fusion strategies, which aim to improve detection performance by capturing diverse and complementary information about the anomaly.

The input feature map $X_{\text{in}} \in \mathbb{R}^{W\times H\times C}$ is first split along the channel dimension into four equal parts: Y_0, Y_1, Y_2, and Y_3. Each part is processed through a distinct branch to extract complementary features. The two bottom branches (Y_0 and Y_1) focus on spatial and contextual information using convolution layers, while the two upper branches (Y_2 and Y_3) adopt the pooling strategies to capture high- and low-frequency features, respectively.

Specifically, Y_0 and Y_1 are each passed through a 3×3 convolution layer to retain detailed spatial features. Y_2 undergoes max-pooling followed by a 1×1 convolution to extract high-frequency information, whereas Y_3 is processed by average pooling and a 3×3 convolution to capture low-frequency context. The outputs from the four branches are then concatenated along the channel dimension to form the final output feature map Y_{out}.

The feature extraction process in the LDM is formally defined as:

$$\begin{aligned} &Y_0, Y_1, Y_2, Y_3 = Split(X_{in}) \\ &O_0 = CBAD_{3\times3}(Y_0) \\ &O_1 = CBA_{3\times3}(Y_1) \\ &O_2 = CBA_{1\times1}(MaxPool(Y_2)) \\ &O_3 = CBA_{3\times3}(AvgPool(Y_3)) \\ &Y_{\text{out}} = Concat[O_0, O_1, O_2, O_3] \end{aligned} \tag{2}$$

where Y_0, Y_1, Y_2 and $Y_3 \in \mathbb{R}^{W\times H\times 0.25C}$ are the four parts separated from the input X_{in}. O_0, O_1, O_2 and $O_3 \in \mathbb{R}^{0.5\ W\times 0.5H\times 0.25C}$ represents the features from different branches. $Y_{\text{out}} \in \mathbb{R}^{0.5\ W\times 0.5H\times C}$ represents the final output feature of the LDM. CBAD denotes a dilated convolution with a dilation rate of 3.

3.5. Aggregation network for feature fusion

As illustrated in Fig. 7, we propose an aggregation module designed to integrate multi-scale features with enhanced contextual sensitivity. The module consists of four parallel branches, each comprising a concatenation operation followed by a Feature Aggregation Module (FAM). The notation within the concatenation function (e.g., Concat[P6]) refers to the specific feature level (e.g., *P6*) from the backbone network to which the operation is linked. This configuration facilitates hierarchical feature fusion, allowing the model to capture anomaly of varying scales and complexities effectively.

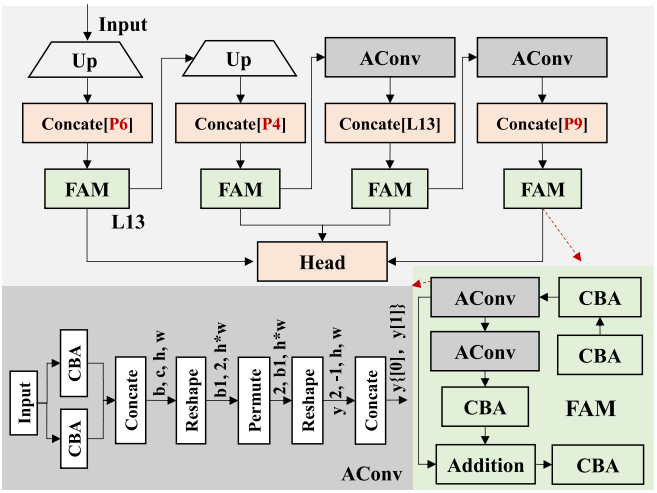


Fig. 7. The graph of the aggregation neck, "Up" and "L13" denote the upsampling operation and the 13th layer of the neural network, respectively.

The Feature Aggregation Module (FAM) contains four stacked CBA blocks (Convolution + BatchNorm + ACLU Function) and two AConv layers, which employ the proposed ACLU activation function. Incorporating additive pathways in deeper layers mitigates the vanishing gradient effect and strengthens feature representation.

The AConv module, depicted as the gray blocks in Fig. 7, is composed of two convolutional layers activated by the ACLU function, followed by a concatenation operation, a reshape layer, and a channel permutation step. To address feature degradation caused by sequential downsampling and convolutional operations, two feature maps are merged early within the AConv block. This early fusion helps retain richer feature information before further transformations.

The subsequent operations are designed to restructure the feature channels and promote cross-channel interaction, thus maintaining strong feature representation while controlling the parameter complexity (space complexity).

After the first reshape operation, the intermediate tensor shape becomes $b_1 = b \times \frac{c}{2}$. The following 'permute' operation swaps the first and second channel dimensions. Assuming that the input to the second reshape operation is denoted as $F \in \mathbb{R}^{2\times b_1\times h\times w}$, the associated computation steps can be formally defined as:

$$\begin{aligned} &y = Reshape(2, t, c, h, w) \\ &F_{out} = Concate(y[0], y[1]) \end{aligned} \tag{3}$$

where F_{out} denotes the final output of AConv. The parameter t can be derived from the other parameters (c, h, w). Similarly, Fig. 8 (third column) indicates that the aggregation neck improves the quality of feature representations, yielding clearer and more discriminative anomaly feature responses.

3.6. Design of the ACLU activation function

In deep neural networks (DNNs) comprising multiple nonlinear hidden layers, an input vector $x_{\text{in}} \in \mathbb{R}^d$ undergoes a series of nonlinear transformations across successive layers. This process can be mathematically formulated as:

$$\begin{aligned} &\beta^0 = x_{\text{in}}, \\ &h_a^{i+1} = \textstyle\sum_{b=1}^{M^l} \omega_{ab}^i \cdot \beta^0 + \alpha_a^i, \\ &\beta_a^{i+1} = \phi(h_a^{i+1}), \end{aligned} \tag{4}$$

where β denotes the activation at a given layer, h_a^i is the pre-activation output, ω_{ab}^i and α_a^i represent the weight and bias parameters respectively, and M^l is the number of units in the ith hidden layer. The function $\phi(x)$ is the activation function applied to each layer's output.

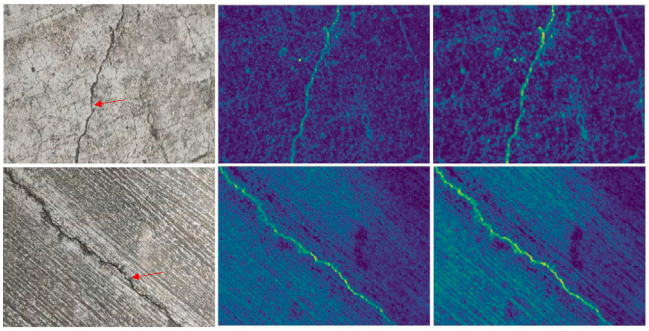


Fig. 8. The figure demonstrates that the aggregation network enables more focused representation of target features, thus improving feature saliency.

Designing flexible and efficient activation functions remains a key challenge in deep network training, since conventional functions such as *Sigmoid* and *Tanh* are prone to saturation, where the derivative approaches zero as $x \to \pm\infty$, resulting in the vanishing gradient and hindered convergence. Although the ReLU function, defined as $\phi(x) = \max(0, x)$, mitigates this issue, it introduces gradient sparsity for negative inputs and the "dying neuron" phenomenon. Despite its widespread adoption, the SiLU activation function exhibits limitations in convergence stability and nonlinear representational capacity, potentially hindering reliable and efficient learning in safety-critical applications.

To address these limitations, we propose a parameterized monotonic activation function, termed ACLU (Adjustable Activation Linear Unit). As illustrated in Fig. 9(a) and (b), ACLU preserves the unbounded behavior on the positive side, similar to the SiLU function, which is widely adopted in recent deep learning architectures such as YOLO [28–30]. The proposed activation function is formally defined as:

$$\phi(x) = a \cdot x \cdot \arctan(e^{bx}), \tag{5}$$

where a and b are learnable or manually-tuned scaling parameters that control the contour of the activation curve.

The ACLU function is designed to balance *nonlinear representation capability* and *gradient stability*, both of which are essential for reliable learning in safety-critical systems. Its construction is motivated by the following considerations:

- Gradient flow: The linear term x ensures effective gradient propagation, particularly for large positive inputs, thereby mitigating vanishing gradient issues typically encountered with saturated activation functions.
- Nonlinear flexibility: The $\arctan(e^{bx})$ term introduces a smooth, bounded nonlinearity that is both monotonic and gradually increasing, facilitating robust convergence.
- Parametric adaptability: The parameters a and b enable dynamic adjustment of the function's shape, allowing it to adapt to different data distributions and training conditions, thereby improving generalization.

As shown in Fig. 9(a), varying the parameters a and b modifies the activation curve: the parameter a adjusts vertical scaling (affecting upper/lower bounds), while b controls the curvature intensity around the origin. Notably, larger values of b tend to flatten the central portion of the curve. In the following, we analyze commonly used activation functions, including LogSigmoid and SiLU, as well as more recent ones such as NELU, Bi-SiLU, and TinyReLU. These functions can be formulated as follows:

$$\text{LogSigmoid}(x) = \log\left(\frac{1}{1+e^{-x}}\right) \tag{6}$$

$$\text{SiLU}(x) = \frac{x}{1+e^{-x}} \tag{7}$$

$$\text{NELU}(x) = \begin{cases} x, & x > 0, \\ \dfrac{-0.2}{1+x^2}, & x \le 0, \end{cases} \tag{8}$$

$$\text{Bi-SiLU}(x) = x \cdot \sigma(x) - 0.835 \tag{9}$$

$$\text{TinyReLU}(x) = \begin{cases} x - e^{-x} + 1, & x \ge 0, \\ 2\,(e^{x} - 1), & x < 0, \end{cases} \tag{10}$$

ACLU vs. LogSigmoid in Fig. 9(b): LogSigmoid is strictly non-positive and saturates to 0^- as $x \to +\infty$, producing vanishing gradients on the positive tail and a persistent negative output bias, which is undesirable as a primary activation in regression/detection heads. By contrast, the ACLU is *zero-crossing* and smooth; it retains a linear positive tail with a tunable slope and applies soft suppression (rather than hard saturation) on the negative side, which improves gradient flow and reduces bias.

ACLU vs. SiLU and TinyReLU in Fig. 9(c): $\text{SiLU}(x) = x\,\sigma(x)$ is smooth with a fixed shape: it softly suppresses negatives and is asymptotically linear for positives, but its curvature and slope are not tunable, which can limit convergence control in very deep stacks. TinyReLU strengthens near-zero responses and imposes a hard negative plateau (and a derivative jump at $x = 0$), which may amplify aliasing or optimization instability. ACLU preserves smoothness and linear positive behavior while providing two degrees of freedom (a, b) to shape curvature and gradient magnitude; it also avoids TinyReLU's non-smooth kink and hard negative saturation, yielding cleaner gradients and more stable training.

ACLU vs. NELU and Bi-SiLU in Fig. 9(d): NELU is piecewise (linear for $x > 0$, rational for $x \le 0$) with a fixed positive slope; Bi-SiLU introduces an offset that shifts the response but leaves the positive-side slope essentially fixed. Both offer limited control over curvature and gradient scaling. ACLU, instead, lets one (i) match or exceed a ReLU/SiLU-like positive slope by setting $a \approx 2/\pi$ (or larger if needed) and (ii) tune the negative soft-saturation via b, improving robustness to background noise while maintaining recoverable gradients.

Stability and convergence: As shown in Fig. 10, we report the validation accuracy and loss trajectories for ACLU, LogSigmoid, SiLU, NELU, Bi-SiLU, and TinyReLU. The ACLU consistently achieves higher accuracy across epochs while maintaining lower validation loss, indicating superior stability and more reliable convergence. Notably, Bi-SiLU exhibits an abrupt accuracy collapse around epoch 400, accompanied by a sharp increase in loss, revealing a sudden training instability that leads to pronounced degradation in performance.

3.7. Loss function

Our methodology employs two distinct loss functions: (i) Classification Loss, which assesses the accuracy of the predicted class for the detected object, and (ii) Regression Loss , which evaluates the precision of the predicted bounding box coordinates relative to the ground truth object position.

3.7.1. Classification loss

As outlined in Eq. (11), the classification loss is computed using the binary cross-entropy method:

$$\begin{aligned} L_{cls} &= -g\log(\sigma(p)) + (1-g)\log(1-\sigma(p)), \\ \sigma(p) &= \tfrac{1}{1+e^{p}}. \end{aligned} \tag{11}$$

where g and p denote the ground truth and predicted class probabilities, respectively.

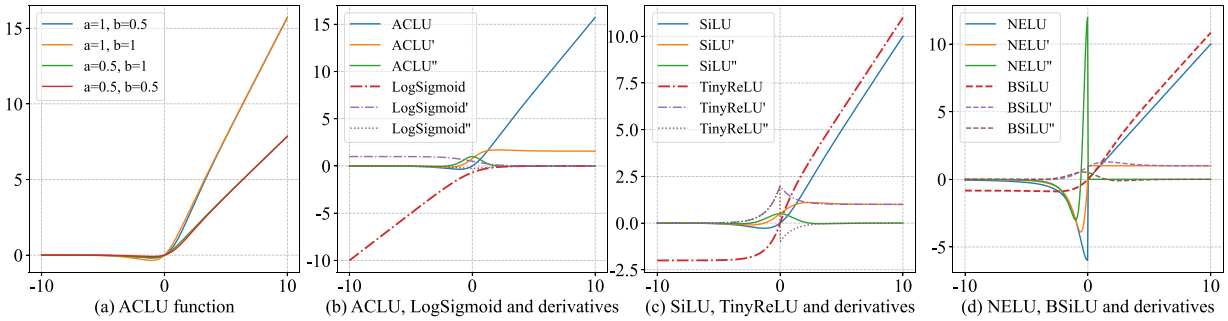


Fig. 9. The plots of the functions. (a) Graphs of the ACLU function for different values of a and b. (b) Plots of the ACLU and LogSigmoid functions, including their first and second derivatives. (c) Plots of the SiLU and TinyReLU functions, along with their first and second derivatives. (d) Plots of the NELU and BSiLU functions, including their first and second derivatives.

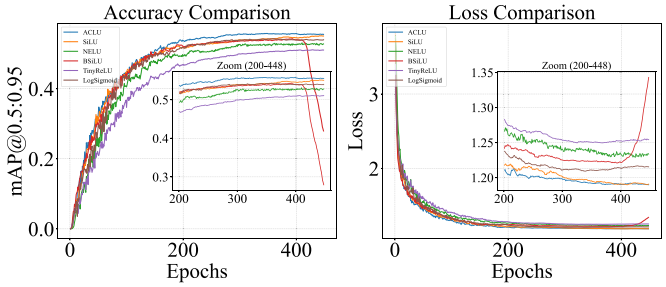


Fig. 10. The left plot illustrates the progression of accuracy, while the right plot shows the corresponding loss. The inset plots can provide a clearer view of the differences in convergence behavior during the training phase.

Table 1
Detailed configuration of our method for training.

Hyperparameter	Value	Hyperparameter	Value
Epochs	500	DFL loss gain	1.5
Optimizer	auto	HSV hue augmentation	0.015
lr0	0.01	HSV saturation augmentation	0.7
lrf	0.02	HSV value augmentation	0.4
lr decay	Linear	Translation augmentation	0.1
Momentum	0.937	Scale augmentation	0.5
Weight decay	0.0005	Mosaic augmentation	1.0
Warmup epochs	3.0	Mixup augmentation	0.0
Warmup momentum	0.8	Paste augmentation	0.0
Warm up bias learning rate	0.1	Close mosaic epochs	10
Box loss gain	7.5	Hypergraph threshold	6
Class loss gain	0.5	Seed	0
Pretrained	False	Erasing	0.4
Crop fraction	1.0	Copy paste mode	Flip

3.7.2. Regression loss

The regression loss comprises two components: L_{DFL} and L_{iou}. The term L_{DFL} enhances the model's generalization capability and is formulated as:

$$\begin{aligned} L_{DFL} &= -((y_{i+1} - y)\log(s_i) + (y - y_i)\log(s_{i+1})), \\ s_i &= \frac{y_{i+1}-y}{y_{i+1}-y_i}, \\ s_{i+1} &= \frac{y-y_i}{y_{i+1}-y_i}, \end{aligned} \tag{12}$$

where y_i and y_{i+1} represent the discrete values bounding the ground truth y, and s_i and s_{i+1} are the corresponding softmax-normalized probabilities.

The regression loss is further defined as:

$$\begin{aligned} L_{ciou} &= 1 - IOU + \frac{Dis^2(b,\hat{b})}{c^2} + \rho k, \\ IOU &= \frac{|B \cap \hat{B}|}{|B \cup \hat{B}|}, \\ \rho &= \frac{k}{(1-IOU)+k}, \\ k &= \frac{4}{\pi^2}\left(\arctan\frac{\hat{\omega}}{\hat{h}} - \arctan\frac{w}{h}\right)^2, \end{aligned} \tag{13}$$

In this formulation, $\hat{B} = (\hat{x}, \hat{y}, \hat{\omega}, \hat{h})$ represents the ground truth bounding box coordinates, while $B = (x, y, \omega, h)$ corresponds to the predicted bounding box. The variables b and $\hat{b}$ denote the centers of B and $\hat{B}$, respectively. The function $Dis(\cdot)$ computes the Euclidean distance between these two centers, and c represents the diagonal length of the smallest enclosing box that contains both B and $\hat{B}$.

The total loss is a weighted combination of the classification and regression losses, expressed as:

$$Loss_{total} = 0.5L_{cls} + 1.5L_{dfl} + 7.5L_{iou}. \tag{14}$$

This formulation ensures a balanced optimization of both classification accuracy and bounding box localization precision during training.

4. Experiment

4.1. Experimental setup

4.1.1. Implementation details

All experiments are conducted using a combination of 1 GPU of type GTX 3090 and 2 GPUs of type GTX 4090D. The working environment is configured with PyTorch 1.8.1 and Python 3.9 on an Ubuntu 20.04 operating system. Automatic Mixed Precision (AMP) training is employed to enhance computational efficiency. During the training process, Mosaic data augmentation is applied to improve the robustness and generalization of the model. Detailed hyperparameter settings are provided in Table 1. Under the "auto" setting, the optimizer is selected by the expected training steps: SGD for more than 10,000 iterations and Adam otherwise.

4.1.2. Datasets

To demonstrate the generalization performance of DEANet, we evaluate the proposed framework on three challenging datasets: CUBIT [48], SSGD [49], and NEU-DET [50]. CUBIT [48] is a drone- and robot-collected anomaly dataset containing high-resolution images (8000 × 6000 and 4624 × 3472). To ensure resolution consistency, we excluded the few 8000 × 6000 images (primarily depicting cracks), which had minimal impact on the dataset's overall composition. The dataset includes cracks, spalling, and moisture in an approximate ratio of 25:5:1, resulting in a diverse range of crack samples but relatively few moisture examples, thereby limiting the model's ability to fully learn moisture-related features.

4.2. Comparison with prior works

CUBIT. A total of 20 state-of-the-art methods were trained for comparative analysis. As shown in Table 2, the quantitative results demonstrate significant improvements in both detection accuracy and computational cost (BFLOPs). Our method achieves the highest ($mAP_{0.5:0.95}$) of 56.2%, while requiring only 2.8 million parameters and 16.6 BFLOPs of computation cost.

Table 2
Quantitative benchmarking results on CUBIT.

Model	Backbone	mAP_{all} (%)↑		mAP_{crack} (%)↑		$mAP_{spalling}$ (%)↑		$mAP_{moisture}$ (%)↑		Parameters (M)↓	FLOPs (B)↓
–	–	$mAP_{0.5}$	$mAP_{0.5:0.95}$	$mAP_{0.5}$	$mAP_{0.5:0.95}$	$mAP_{0.5}$	$mAP_{0.5:0.95}$	$mAP_{0.5}$	$mAP_{0.5:0.95}$	–	–
Faster R-CNN [31]	ResNet50	69.4	41.7	68.4	37.8	83.9	53.8	55.9	33.5	41.8	220.2
YOLOv6-n [32]	RepBlock	76.5	49.8	77.8	48.4	87.3	60.8	57.6	31.7	5.2	29.0
PVT-FPN [33]	PVT-t	63.5	34.3	53.2	24.5	79.0	48.0	58.4	30.5	23.0	225.4
YOLOv7 [34]	ELAN-Net	77.2	49.7	80.8	49.0	87.6	59.8	63.0	40.3	39.3	264.3
RetinaNet [35]	ResNet18	58.5	30.4	51.1	23.8	73.0	41.4	51.4	26.0	21.4	197.9
Casecade R-CNN [36]	ResNet50	67.4	39.6	63.0	34.2	81.0	51.2	58.2	33.4	69.4	256.9
RT-DETR-l [37]	HGNetv2	78.4	48.9	77.8	45.5	85.7	60.7	71.8	40.6	34.7	264.7
RT-DETR-x [37]	HGNetv2	79.2	49.9	79.2	46.5	86.0	60.6	72.4	42.7	71.0	569.6
RT-DETR-ResNet50 [37]	ResNet50	71.9	45.8	73.6	40.7	86.6	60.2	55.4	36.5	41.9	321.5
RT-DETR-ResNet101 [37]	ResNet101	73.0	46.1	73.8	42.0	89.1	61.3	41.7	35.2	60.9	476.7
YOLOv9-s [38]	GELAN	79.2	49.9	79.2	46.5	86.0	60.6	72.4	42.7	7.1	68.4
FCOS [39]	ResNet50	58.7	31.8	53.8	25.7	70.4	41.8	52.0	27.8	32.3	209.7
EfficientDet [40]	Effficientb3	12.4	5.3	22.7	9.5	8	3.7	6.6	2.6	20.0	178.3
YOLOv10 [29]	CSPNet	79.5	53.3	81.4	52.9	90.6	64.8	66.5	42.3	16.5	151.3
D-FINE [41]	HGNetv2	73.0	48.8	–	–	–	–	–	–	4.0	17.9
Hyper-YOLO [42]	–	78.8	51.5	81.5	50.7	88.7	61.5	66.3	42.3	3.0	22.8
LSM-YOLO [43]	–	76.2	49.3	74.7	44.9	89.5	61.6	64.3	41.5	2.9	32.3
FCM [44]	–	70.8	42.9	72.0	40.0	84.1	53.5	56.2	35.1	2.9	58.6
MHAF-YOLO [45]	–	80.2	54.4	83.6	54.1	88.3	64.7	68.7	44.4	7.4	68.4
DCAS [46]	–	72.4	44.6	73.3	41.8	84.3	54.7	59.4	37.3	4.1	85.5
DEANet (Ours)	–	**80.7**	**56.2**	**83.5**	**54.7**	**91.5**	**67.8**	**67.9**	**46.0**	**2.8**	**16.6**
DEANet-s (Ours)	–	**81.4**	**56.1**	**82.5**	**54.4**	**91.9**	**67.2**	**69.8**	**46.7**	**2.8**	**16.6**

The parameters of activation function in DEANet are $a = 0.6$ and $b = 0.42$. DEANet-s refers to the method of utilizing SDIOU [47] for bounding box regression.
↑ (↓) indicates that larger (smaller) values lead to better (worse) performance.

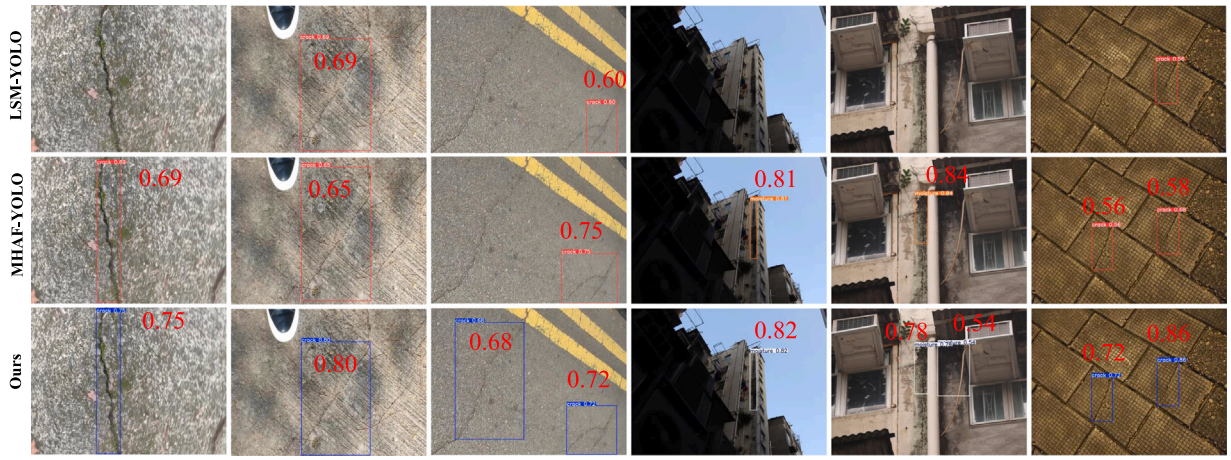


Fig. 11. Our method is compared qualitatively with the new methods, LSM-YOLO and MAHF-YOLO. Our method exhibits fewer missed detections and more accurate localization (third row), even under low-light conditions (fourth and sixth columns). Please zoom in for the best view.

Compared to the LSM-YOLO model, our approach reduces computational cost by 48.6% and simultaneously improves accuracy by 14.0%. Relative to accurate MHAF-YOLO, our DEANet still achieves a 75.7% reduction in computational cost and a 62.2% reduction in parameter count, along with a noticeable performance gain. These results underscore the proposed method's ability to achieve a favorable trade-off between accuracy and resource consumption, making it particularly suitable for deployment in resource-constrained settings.

Fig. 11 provides visual comparisons of detection outputs from our method, LSM-YOLO, and MHAF-YOLO. As shown, our model consistently yields higher confidence scores and fewer missed detections, indicating enhanced robustness and reliability in practical scenarios. Notably, our method continues to perform well under low-light conditions, as illustrated in the fourth and sixth columns.

Furthermore, Fig. 1 illustrates the superior trade-off achieved by our method with respect to both computational cost and detection accuracy, as well as parameter count and overall performance. Collectively, these results confirm the effectiveness and practicality of the proposed approach for real-world, deep learning-based anomaly detection tasks, particularly in domains where computational resources are limited.

We further evaluated the few-shot learning capability of the proposed method, as illustrated in Fig. 12. The left plot shows that our approach achieves accuracy comparable to PVT-FPN while substantially reducing resource demands, requiring 70% fewer training samples, 87.8% fewer parameters, and 92.6% less computational cost. Compared with recent state-of-the-art methods such as Hyper-YOLO and D-FINE, it further reduces the required training data by 20%, confirming its robustness in data-scarce settings.

The right panel of Fig. 12 also presents accuracy trends across three anomaly categories under varying training sample ratios. Notably, for the moisture category (which includes relatively few training instances), the rate of accuracy improvement plateaus once the training data exceeds 30%. In some cases (70 vs. 60; 40 vs. 30), performance with limited data even surpasses that achieved using the larger dataset, indicating that the method effectively leverages contextual information to improve detection accuracy in low-data scenarios.

While the method performs robustly under most configurations, the results also suggest that extremely limited data may constrain the capacity of direct feature extraction; for example, when only 10% of the training data are used, the detection accuracy for Moisture drops

Table 3
Quantitative benchmarking results on SSGD.

Method	Parameters (M)↓	FLOPs (B)↓	FPS↑	mAP_1 (%)↑	mAP_2 (%)↑
Faster R-CNN [31]	41.2	303.8	26.2	19.3	41.5
Casecade R-CNN [36]	68.9	331.6	21.7	20.9	42.3
RetinaNet [35]	36.2	311.2	25.0	16.4	37.5
FCOS [39]	31.9	296.2	28.1	19.4	41.9
ATSS [51]	31.9	303.3	24.2	22.3	46.1
GFL [52]	32.1	307.9	25.0	19.6	43.2
YOLOv5-m [27]	19.9	266.7	59.5	16.2	38.9
YOLOX-m [53]	48.3	405.1	36.9	13.4	36.2
YOLOv8-m [28]	27.4	432.3	71.8	21.7	46.2
Swin-T [54]	44.8	308.2	18.1	19.2	42.6
PVT-S [33]	78.4	281.3	12.3	16.0	36.7
ScalableViT-S [55]	43.3	297.7	10.9	21.2	46.4
UniFormer-Sh14$_{hl4}$ [56]	38.2	276.4	15.8	18.9	45.0
YOLOv9-t [38]	3.3	60.4	111.1	22.2	45.5
YOLOv10 [29]	16.5	351.6	50.3	10.3	29.8
LSM-YOLO [43]	2.9	69.2	142.9	16.6	38.6
FCM [44]	2.9	125.8	149.0	17.5	40.3
MHAF-YOLO [45]	7.4	146.7	52.4	13.8	32.1
DCAS [46]	4.1	183.5	147.0	14.3	32.9
DEANet(1, 0.7)	**2.8**	**37.7**	**153.9**	**24.9**	**52.0**
DEANet(1, 0.5)	**2.8**	**37.7**	**153.9**	**21.9**	**56.2**

mAP_1 and mAP_2 represent $mAP_{0.5:0.95}$ and $mAP_{0.5}$, respectively. The parameters in parentheses represent the values of a and b in the ACLU.

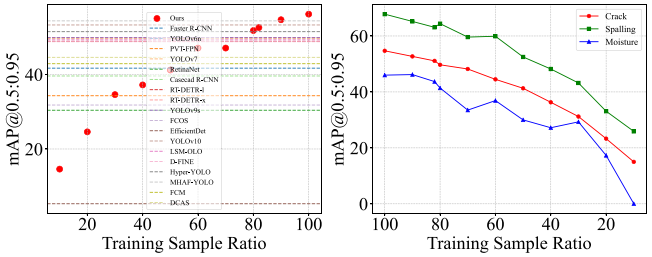


Fig. 12. Left: Comparison of accuracy performance under different sample proportions during training. Right: Accuracy variation of different target types under different training sample proportions. Please zoom in for the best view.

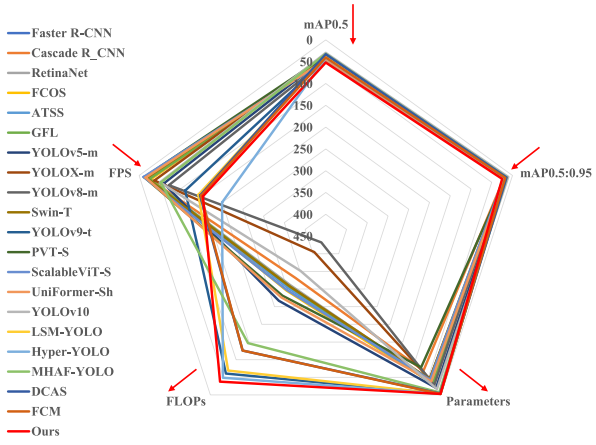


Fig. 13. Comparison of balanced performance across five evaluation metrics. The closer a point is to the direction indicated by the arrow, the better the corresponding metric. For example, for $mAP_{0.5}$, the arrow points inward, indicating that higher values (closer to the center) are better. In contrast, for the parameter count, the arrow points outward, indicating that lower values (those farther from the center) are preferred.

to nearly zero. These findings highlight the complementary roles of explicit visual features and context-aware modeling in few-shot anomaly detection, offering insights for further optimization under severe data constraints.

Overall, the results highlight the proposed method's robustness and training data efficiency in few-shot learning scenarios, particularly in real-world settings where annotated data is limited or expensive to obtain.

SSGD. Similarly, as shown in Table 3, the proposed method consistently outperforms competing approaches across five evaluation metrics. In the table, the upper block (above the YOLOv9-t demarcation) reports results from the original paper [49], whereas the remaining entries correspond to our own training. Compared with YOLOv9-t, which has a comparable parameter count, our method achieves higher accuracy (**24.9** vs. 22.2), faster inference speed (**153.9** vs. 111.1 FPS), and significantly lower computational cost (**37.7** vs. 60.4 BFLOPs). When compared to the state-of-the-art YOLOv10, our method reduces parameter size by 83% and computational load by 89.3%, while nearly doubling accuracy and delivering a substantial improvement in inference speed.

In addition, our method outperforms the recently proposed LSM-YOLO and DCAS, achieving higher accuracy with significantly lower computational requirements. Compared to MHAF-YOLO, it reduces the number of parameters by 62.2% and the computational load by 74.3%, while nearly tripling the inference speed and doubling the accuracy.

Fig. 13 further visualizes the performance distribution using a radar chart across five evaluation dimensions, normalized to a common scale (0–450). For reference, YOLOv8-m exhibits the highest computational cost in Table 3, with 432.3 BFLOPs.

These results collectively demonstrate that our method achieves a highly favorable balance between detection accuracy, speed, and resource efficiency. This trade-off is fundamental in real-world applications, where computational constraints and performance requirements often conflict. By significantly reducing resource consumption while maintaining or improving performance, the proposed approach offers a scalable and deployable solution for anomaly detection in resource-constrained environments.

Finally, the detection visualizations in Fig. 14 further validate the practical effectiveness and robustness of our method across diverse anomaly types and scenes.

NEU-Det. For subsequent training, we selected new models that achieved favorable performance on the first two datasets while maintaining low computational requirements. For our comparative analysis in Table 4, we selected Hyper-YOLO, a state-of-the-art method, which utilizes hypergraph convolutions as its feature extraction module. Our proposed approach, DEANet, achieves superior accuracy while reducing the parameter count by 7% compared to Hyper-YOLO.

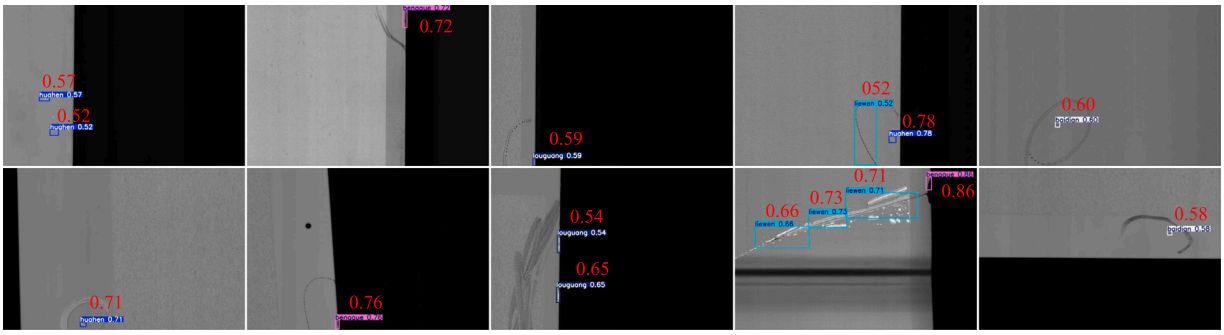


Fig. 14. The detection visualization on the SSGD dataset shows that, although the defects in SSGD are relatively small, our method is still able to detect them successfully.

Table 4
Quantitative benchmarking results on NEU-Det.

Method	$mAP_{0.5}$	Crazing	Inclusion	Rolled	Scratches	Pathes	Pitted
RetinaNet [35]	68.0	43.7	76.2	58.1	76.0	74.3	79.6
LSM-YOLO [43]	73.2	39.3	79.9	56.5	91.6	91.7	80.6
Hyper-YOLO [42]	74.9	40.7	82.8	59.3	92.8	91.9	81.6
FCM [44]	72.9	46.2	77.2	52.3	91.7	91.0	78.8
MHAF-YOLO [45]	71.1	39.0	75.7	51.5	89.4	89.0	82.2
DCAS [46]	71.6	40.4	79.2	51.2	91.2	91.0	77.3
DEANet	**76.5**	**45.0**	**84.4**	**59.9**	**94.8**	**92.7**	**82.4**

The bold values represent the experimental results from the proposed method.

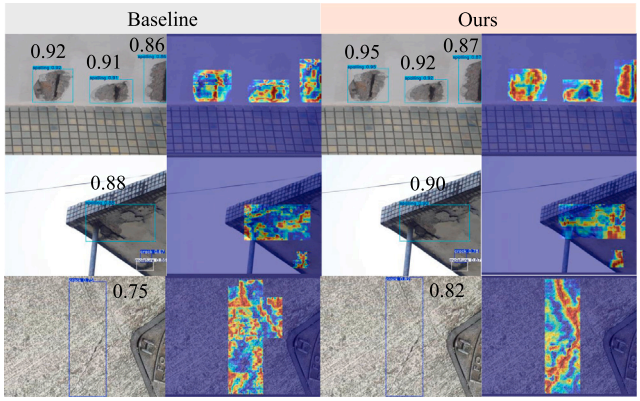


Fig. 15. This figure illustrates that our method effectively isolates and extracts salient target features while disregarding redundant or non-informative features. Additionally, our method exhibits a larger receptive field in the feature extraction phase when detecting moisture.

4.3. Ablation study and discussion

To assess the individual contributions of the proposed components, we conduct comprehensive ablation experiments on the CUBIT dataset. All evaluations are performed with a fixed input resolution of 1024 × 1024 during the inference phase. We adopt YOLOv11, a widely used single-stage object detection model, as the baseline for comparison.

As shown in Table 5, each proposed module, namely anomaly detection and scale estimation, contributes to a notable improvement in detection accuracy. Integrating the two modules yields a substantial performance gain, underscoring the effectiveness of our approach in addressing the challenges associated with anomaly variability and multiscale object representation.

These results highlight the crucial role of each component in achieving state-of-the-art accuracy while maintaining computational efficiency. Furthermore, Fig. 15 visually confirms the superiority of our method over the baseline, with enhanced feature extraction and more accurate anomaly localization.

4.3.1. Benefits of aggregation neck

The Aggregation Neck is designed to integrate multi-scale features and enhance the visibility of features more effectively (See Fig. 8), thereby improving the model's representational capacity while maintaining computational efficiency. As shown in Table 5, this module leads to a nearly 3% improvement in the $mAP_{0.5:0.95}$ metric, indicating superior performance under stringent evaluation criteria.

In addition to improved detection accuracy, the Aggregation Neck reduces the parameter count and computational cost by 19% and 11%, respectively. These results demonstrate that the proposed aggregation neck enhances multi-scale feature fusion and significantly improves model efficiency, making it particularly suitable for deployment in resource-constrained environments.

4.3.2. Benefits of CFM

As shown in Table 5, integrating the proposed CFM leads to improvements of 1.3 and 0.2 percentage points in $mAP_{0.5}$ and $mAP_{0.5:0.95}$, respectively, compared to the baseline integrated aggregation neck. The CFM employs a dual-branch convolutional structure to separately extract anomaly features and environmental context information, thereby enabling a more robust feature representation.

While this design introduces a modest increase in parameters and computational cost, it significantly enhances contextual awareness, particularly benefiting anomaly types that are closely associated with environmental factors. For instance, moisture often occurs under humid conditions and exhibits spatial correlation with surrounding textures. Expanding the receptive field through CFM enables the model to capture contextual cues better, thereby improving localization accuracy.

Table 6 further highlights this advantage: the detection performance for cracks and spalling remains stable, while the $mAP_{0.5}$ and $mAP_{0.5:0.95}$ for moisture improve by 6.0% (i.e., $(73.4-69.6)/69.6\times100\%$) and 1.1%, respectively, with the integration of the CFM module. These results validate the effectiveness of CFM in enhancing detection performance under context-dependent scenarios.

Fig. 16 qualitatively compares detection results with and without the CFM module under simple (top two rows) and complex (bottom two rows) backgrounds. The results with CFM (second and fourth rows) exhibit fewer missed detections and higher confidence, demonstrating the performance gain of the proposed module.

4.3.3. Benefits of DEM

The DEM is designed to simultaneously preserve high-frequency and low-frequency information during the downsampling process, thereby mitigating the performance degradation typically caused by information loss. In addition, the DEM can enhance feature discriminability by suppressing redundant features (See Fig. 5). This enriched feature

Table 5
Effects of various components on performance.

Components	Ablation study					
Aggregation neck		✓	✓	✓	✓	✓
CFM			✓	✓	✓	✓
DEM				✓	✓	✓
LDM					✓	✓
ACLU						✓
mAP_1/mAP_2 (%)	53.4/80.3	55.0/80.0	55.2/81.6	55.6/80.1	56.0/80.5	56.2/80.7
Improvement	–	+1.6/−0.3	+1.8/+1.3	+2.2/−0.2	+2.6/+0.2	+2.8/+0.4
Parameters (M)	2.62	2.12	2.32	2.80	2.76	2.76
Improvement	–	−0.5	−0.3	+0.18	+0.14	+0.14
FLOPs (B)	6.6	5.9	6.4	7.0	6.5	6.5
Improvement	–	−0.7	−0.2	+0.4	−0.1	−0.1

The ✓denotes that the module is integrated into this ablation study.
The improvement is measured relative to the baseline.

Table 6
Ablation study on CFM.

Method	mAP_{all} (%)↑		mAP_{crack} (%)↑		$mAP_{spalling}$ (%)↑		$mAP_{moisture}$ (%)↑	
–	mAP_2	mAP_1	mAP_2	mAP_1	mAP_2	mAP_1	mAP_2	mAP_1
w/o CFM	80.0	55.0	81.7	53.1	88.8	65.9	69.6	45.9
w/ CFM	**81.6**	**55.2**	**81.6**	**53.1**	**89.9**	**65.9**	**73.4**	**46.5**

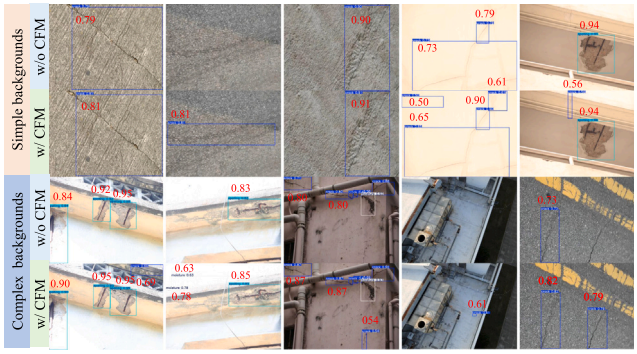


Fig. 16. This figure demonstrates that the proposed CFM module consistently enhances detection performance across both simple and complex backgrounds. The first two rows depict simple backgrounds, and the last two rows depict complex backgrounds, with the second and fourth rows presenting the results obtained after integrating the CFM module.

representation contributes to improved detection accuracy, particularly in object localization.

As shown in Table 5, the introduction of DEM leads to an increase in localization performance from 55.2 to 55.6 $mAP_{0.5:0.95}$. Despite adding only a marginal number of parameters and minimal computational overhead, DEM demonstrates its effectiveness as a lightweight and efficient enhancement to the detection framework.

4.3.4. Benefits of LDM

To further optimize DEANet, we introduce an LDM that operates in conjunction with the DEM. The LDM is designed to capture multi-level features of the anomaly by integrating shallow and deep semantic information. Given that both low-level and high-level features may be incomplete when extracted in isolation, the LDM is strategically placed at intermediate stages to enhance representation through joint utilization with DEM.

As shown in Table 5, the incorporation of the LDM leads to a reduction of 0.04M parameters and 0.5 BFLOPs in computational cost, while simultaneously improving detection accuracy from 55.6 to 56.0 in $mAP_{0.5:0.95}$. These results demonstrate that the LDM enhances performance and contributes to a more computationally efficient architecture.

Table 7
Ablation study of LDM.

Position	$mAP_{0.5:0.95}$ (%)↑	$mAP_{0.5}$ (%)↑
1, 2	54.3	79.4
1, 2, 3	54.3	78.5
2, 3	55.0	80.4
1	**56.0**	**80.5**

We also conduct experiments by replacing different DEM modules with the LDM. Before integrating the LDM, the DEM exists in the P4 and P6 stages, as shown in Fig. 2. Since P6 and P7 are repeated twice, the DEM appears at three positions. These positions are labeled from left to right as positions 1, 2, and 3. For example, the second row in Table 7 indicates that the LDM module exists at positions 1, 2, and 3. The last row of experimental results demonstrates that applying the LDM at the first position, which corresponds to the intermediate stage of feature extraction, leads to a significant improvement in detection performance. Lower layers primarily capture edge and contour cues, whereas higher layers encode abstract semantics. Intermediate layers provide information-rich representations that integrate spatial detail with contextual cues, thereby enabling more multifaceted semantic modeling.

4.3.5. Benefits of ACLU

By tuning two specific parameters within ACLU, the function can effectively stimulate the nonlinear representational capabilities across diverse neural network architectures. As demonstrated in Table 5, ACLU enhances the model's nonlinear representational power without increasing the number of parameters. The use of ACLU further improves the model's $mAP_{0.5:0.95}$ performance by 2.8 percentage points. This also demonstrates that the proposed ACLU activation function outperforms SiLU in terms of accuracy.

Sensitivity analysis: We conduct a comprehensive sensitivity study of the ACLU's parameters on CUBIT, SSGD, and NEU to assess their robustness and performance under both learnable and manually settings. Given that the parameter a primarily controls the vertical amplitude of the ACLU curve, while b governs its nonlinearity, our tuning experiments focus on analyzing the sensitivity of activation performance with respect to b. The results are summarized in Table 8. Experimentally, optimal accuracy is attained for $b \in [0.3, 1.0]$.

Table 8
Parameters sensitivity analysis on ACLU.

a (CUBIT)	learnable	0.8	1	1	1	1	1	1	1	1	0.6	1.2
b (CUBIT)	Learnable	0.42	0.51	0.3	0.4	0.5	0.55	0.6	0.7	0.8	0.42	0.42
$mAP_{0.5:0.95}$	47.1	55.7	55.7	54.4	55.4	55.1	55.8	54.9	55.0	54.6	**56.2**	54.6
$mAP_{0.5}$	73.8	80.5	81.0	79.6	80.1	80.4	**81.2**	80.4	81.2	79.8	80.7	81.5
a (SSGD)	Learnable	0.8	1	1	1	1	1	0.6	1	1	1	1
b (SSGD)	Learnable	0.5	0.75	0.65	0.4	0.5	0.6	0.42	0.7	0.8	0.9	1
$mAP_{0.5:0.95}$	20.1	20.7	24.5	20.3	22.3	21.9	19.0	22.9	**24.9**	18.6	21.0	21.3
$mAP_{0.5}$	46.4	44.2	50.1	47.4	46.7	**56.2**	46.5	48.8	52.0	42.5	45.8	48.3
a (NEU)	Learnable	1	1	1	1	1	1	1	1	1	1	2
b (NEU)	Learnable	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0	1.0
$mAP_{0.5:0.95}$	43.9	42.0	43.9	44.1	44.4	43.3	43.8	44.1	44.5	44.0	43.8	**44.7**
$mAP_{0.5}$	74.2	72.7	74.7	**76.5**	75.8	74.9	75.7	75.4	76.0	75.7	75.5	75.6

This experiment is conducted to identify the optimal values for parameters a and b.

Table 9
Activation function comparison.

Activation function	$mAP_{0.5:0.95}$ (%)↑	$mAP_{0.5}$ (%)↑	Computational cost
Softplus	54.5	79.2	High
RReLU	53.2	78.8	Low
LeakyReLU [21]	53.7	79.2	Low
Tanh	48.7	73.3	High
PReLU [57]	53.9	80.8	Low
CELU [58]	53.3	78.4	Medium
LogSigmoid	54.2	78.8	High
TanhShrink	27.3	48.4	High
Hardtanh	47.4	73.7	Low
ELU [23]	53.3	78.4	Medium
SELU [24]	51.0	76.2	High
Sigmoid	48.3	73.4	High
Softsign [59]	40.7	64.1	Medium
Softshrink	41.7	65.4	Low
TinyReLU [9]	51.3	77.0	Medium
NELU [10]	52.9	78.3	low
BSiLU [10]	54.1	79.8	Medium
GeLU [8]	52.5	79.9	High
AGLU [7]	48.8	75.1	High
HardShrink	11.0	23.7	Low
ACLU (ours)	**55.8**	**81.3**	Medium

We adopt a two-stage tuning protocol: first optimize b, which governs the degree of nonlinearity, and then adjust a (typically fixed within $\{1, 2\}$), which scales the activation amplitude. This schedule improves numerical stability by reducing the likelihood of gradient explosion and enhancing the convergence properties. We observe that the learnable configuration attains lower accuracy than manual tuning, likely due to overfitting induced by the additional degrees of freedom introduced by the learnable parameters, which degrades generalization. In addition to the scalar self-learning reported in Table 8, we also conducted experiments on channel-wise self-learning. The key distinction lies in the fact that, for channel-wise self-learning, each channel in a layer is assigned different parameters a and b. We observed that the performance of channel-wise self-learning is inferior to that of scalar self-learning. This is because channel-wise parameterization substantially increases the number of learnable parameters, leading to potential overfitting on limited data

Overall, we recommend a two-stage calibration of ACLU: first tune b (within 0–1), then adjust a (typically 1–2). For rapid prototyping, a learnable activation paradigm can be adopted. Although the learnable variant underperforms the manually tuned configuration on the three datasets considered, we expect its advantages to emerge as the training data scale increases.

Activation performance comparison: Table 9 presents a comparative analysis between ACLU and 20 state-of-the-art activation functions. Across all results, ACLU consistently achieves higher accuracy. In particular, compared to the recently proposed AGLU function, ACLU improves $mAP_{0.5}$ and $mAP_{0.5:0.95}$ by 8.3% and 6.3%, respectively, demonstrating its enhanced nonlinear representational capacity and stronger generalization for complex anomaly scenarios. Compared with TinyReLU, which is specifically designed for anomaly detection, our method achieves absolute gains of 5.6% in $mAP_{0.5}$ and 8.8% in $mAP_{0.5:0.95}$.

In terms of computational complexity, ACLU exhibits moderate resource consumption, as defined in Table 10. Compared to PReLU [57], which achieves the highest accuracy among low-cost activation functions, ACLU improves $mAP_{0.5:0.95}$ by 4.9%. Among activation functions with moderate complexity, CELU attains the highest baseline accuracy; however, ACLU surpasses it with an additional 4.7% gain in $mAP_{0.5:0.95}$. Similarly, Softplus, which achieves high accuracy among high-complexity activation functions, shows a 2.4% decline in $mAP_{0.5:0.95}$ compared to ACLU. This further confirms the superiority of ACLU in activation performance.

Moreover, real-time inference evaluations confirm that ACLU maintains high-speed processing without introducing latency, reinforcing its suitability for deployment in time-critical and resource-constrained anomaly detection applications (See Section 4.4 for details).

4.4. Edge-computing device deployment evaluation

The effectiveness of the proposed detection method has been validated using three publicly available datasets. To further assess its suitability for real-world deployment, DEANet was implemented on a memory-constrained edge-computing platform (Jetson Orin NX) to evaluate its practicality, portability, and lightweight design.

A total of 534 test images were used in the evaluation. The results show that DEANet achieves a real-time inference speed of 52.1 frames per second (FPS) on the edge-computing platform. As illustrated in Fig. 17, the model successfully detects anomalies with high confidence, demonstrating its robustness and efficiency in resource-limited environments. These results further validate the model's potential for deployment in industrial inspection scenarios requiring both speed and accuracy.

4.5. Robustness analysis: Paired t-Tests and CIs

We collected over 3700 challenging anomaly datasets under sunny weather and post-rainy sunny conditions. The dataset was split into 10 disjoint, equally sized subsets for evaluation. The dataset will be publicly available for testing. For each subset, we evaluated three models (Ours, MHAF-YOLO, and DCAS) and obtained paired $mAP_{0.5:0.95}$ scores across subsets. We then ran two-sided paired Student's t-tests comparing Ours with each baseline. We report the mean difference (Δ, in percentage points), t-statistic, degrees of freedom (df = 9), two-tailed p-value, 95% confidence interval (CI) for Δ, and Cohen's d_z (effect size for paired designs). To control for two pairwise comparisons, we additionally applied a Bonferroni correction (α).

Descriptively, Ours achieved 36.81 ± 1.53 mAP, versus 35.09 ± 2.55 for MHAF-YOLO and 33.65 ± 2.27 for DCAS (mean $\pm$ Standard Deviation (SD) across 10 subsets). Paired t-tests showed that Ours significantly outperformed both baselines:

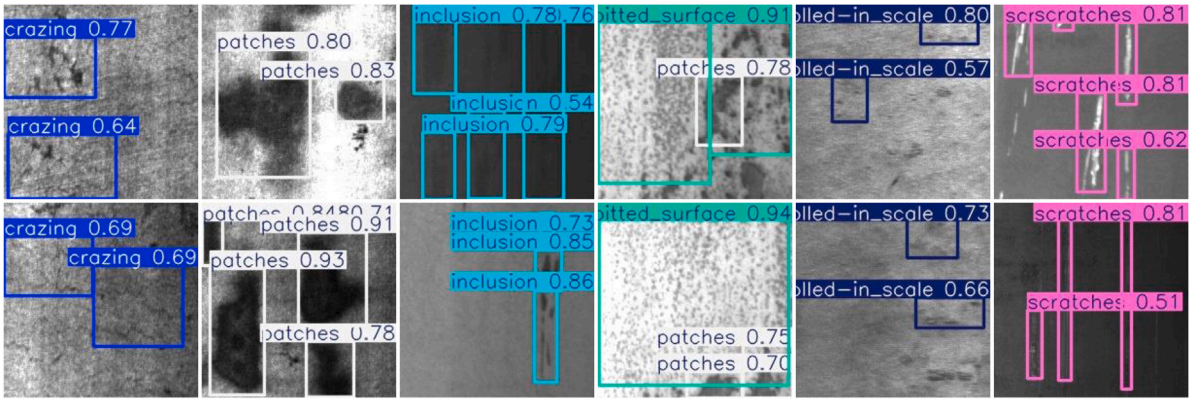


Fig. 17. The detection results on the edge computing device demonstrate that our method can identify anomaly with high confidence.

Table 10
Definition of computational cost levels for activation functions.

Cost level	Description	Typical operations	Examples	Remarks
Low	Involves only basic arithmetic or thresholding operations; highly parallelizable on GPU	Addition, multiplication, max, ReLU-like gating	LeakyReLU, PReLU, Hardtanh, RReLU	HardShrink also belongs here. Efficient single-kernel implementation with minimal overhead.
Medium	Contains one lightweight nonlinear operation; moderately efficient	Single exp, log, or arctan	ACLU (ours), CELU, ELU, Softsign	Moderate runtime. Costlier than ReLU, but suitable for real-time inference.
High	Includes multiple complex functions or hard-to-fuse nonlinearities	exp + log, tanh, erf, division	Sigmoid, GeLU, Softplus, AGLU	TanhShrink is also included. Slow due to compound operations and poor GPU fusion.

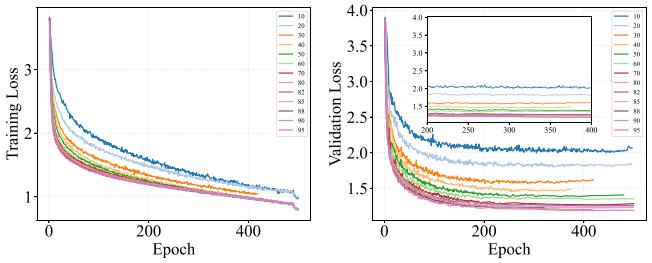


Fig. 18. Training and validation curves under different few-shot settings. The inset on the right illustrates an enlarged view of the training dynamics within the range of 200 to 400 epochs.

Fig. 19. Failure cases of the proposed method.

- Ours vs. MHAF-YOLO: $\Delta = 1.72$ pp, $t(9) = 3.86$, $p = 0.0039$, 95% CI $[0.71, 2.73]$, $d_z = 1.22$.
- Ours vs. DCAS: $\Delta = 3.16$ pp, $t(9) = 6.03$, $p = 0.00020$, 95% CI $[1.97, 4.35]$, $d_z = 1.91$.

Both results remain significant after Bonferroni correction ($\alpha = 0.025$), indicating a reliable improvement of our method over the baselines. The inference latencies of the three methods are 2.1 ms, 5.2 ms, and 3.2 ms, respectively, which further corroborates the efficiency of the proposed method.

5. Limitations and future work

Despite the strong performance demonstrated by DEANet, our method still has certain limitations that offer avenues for future research and enhancement. A primary concern is the potential risk of overfitting when trained on small-scale datasets, which are common in industrial anomaly detection scenarios. This overfitting can compromise the model's generalization ability, particularly when encountering rare or highly specialized anomaly types that are underrepresented in the training data.

As shown in Fig. 18, we present the training curves (left) and validation curves (right) under different levels of few-shot settings, comprising a total of 13 curves. The numbers in the legend indicate the proportion of training samples used; for example, the number 10 denotes that only 10% of the training samples are employed. Training is terminated if the performance remains stagnant or degrades within 100 epochs. From the figure, it can be observed that while the training loss continues to decrease, the validation curve corresponding to 50% of the training samples (dark green) begins to rise around 400 epochs and terminates prematurely, exhibiting an overfitting phenomenon. Future research could explore diffusion-based generative augmentation tailored to industrial anomaly scenarios to expand data diversity or reducing channel dimensionality to achieve model compression, both of which can effectively mitigate overfitting and improve performance in few-shot regimes. Excessive parameterization often results in insufficient feature extraction when training with limited data, thereby yielding suboptimal feature representations and increasing the risk of overfitting.

Although our experiments on the CUBIT, NEU-Det, and SSGD datasets demonstrate robustness across diverse anomaly categories, the

model may still face challenges when dealing with previously unseen or novel anomaly patterns due to the limited size and diversity of available datasets. As shown in Fig. 19, our method also exhibits several failure cases. For instance, missed detections occur in the third column (highlighted with a red dashed box), while false detections appear in the first row of the second column. The first column illustrates inaccurate localization. These issues primarily arise from challenging scenarios, such as low contrast between the target and the background, severe occlusion, or scale variations. Potential strategies for future improvement include incorporating more robust feature representations, introducing adaptive mechanisms for scale and illumination changes, and leveraging multi-modal information or advanced attention modules to enhance model robustness.

In addition, while existing activation functions, including the one proposed in this study, demonstrate favorable activation characteristics, there remains significant room for improving computational efficiency. Notably, when the two tunable parameters of our activation function are set to be learnable, its activation performance tends to degrade. A similar limitation has been observed in AGLU, which involves three learnable parameters but fails to consistently achieve stable improvements. This suggests a need for more robust and efficient parameterization strategies in activation design.

Moreover, DEANet experiences notable performance degradation under adverse environmental conditions such as rain, snow, and fog. Future work will focus on enhancing the model's robustness in such dynamic and complex environments. For instance, in UAV-based deployments, issues such as motion blur and vibration-induced image degradation could be mitigated through image enhancement or stabilization techniques. Additionally, certain component-level anomalies exhibit visual variability based on the severity of structural damage. Future research will focus on severity-aware anomaly classification to enhance risk assessment and facilitate predictive maintenance in safety-critical applications.

6. Conclusion

This paper presents an anomaly detection framework that achieves a strong balance between detection accuracy and computational efficiency. To address the degradation in accuracy caused by downsampling, two complementary downsampling modules were designed to enhance the detection performance. Their joint integration enables a favorable trade-off between accuracy and resource consumption. In addition, we propose a novel activation function that outperforms 20 existing alternatives in terms of accuracy, featuring tunable parameters that enable adaptability across various anomaly detection neural network architectures.

A contextual feature extraction module was incorporated to leverage environmental semantics, particularly enhancing performance under limited sample conditions. Furthermore, a lightweight feature fusion network was introduced, improving accuracy while reducing the parameter count (only 2.8M) and computational load. Experimental results across three benchmark datasets demonstrate that the proposed method significantly improves performance: achieving up to 87.8% reduction in model size, 92.6% lower computational cost, and 70% fewer training samples, without sacrificing accuracy. Real-time inference tests show that our method achieves 52.1 FPS on an edge-computing platform, demonstrating its practicality for deployment in resource-constrained environments.

CRediT authorship contribution statement

Xunkuai Zhou: Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Conceptualization. **Xi Chen:** Writing – review & editing, Supervision, Resources, Conceptualization. **Jie Chen:** Writing – review & editing, Supervision, Resources, Project administration. **Ben M. Chen:** Writing – review & editing, Project administration, Investigation, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was partly supported by the InnoHK initiative of the Innovation and Technology Commission of the Hong Kong Special Administrative Region Government via the Hong Kong Centre for Logistics Robotics and partly by the Research Grants Council of Hong Kong SAR under Grant No. 14217922, 14209623, 14209424 and 14200524.

Data availability

The dataset can be publicly acquired.

References

[1] D. Peng, W. Desmet, K. Gryllias, Reconstruction-based deep unsupervised adaptive threshold support vector data description for wind turbine anomaly detection, Reliab. Eng. Syst. Saf. 260 (2025) 110995.
[2] L. Gao, D. Li, N. Liang, A hybrid sensor fault detection and diagnosis method for air-handling unit based on multivariate analysis merged with deep learning, Adv. Eng. Inform. 65 (2025) 103360.
[3] X. Liao, D. Wang, S. Qiu, M. Xia, X. Ming, SLDAE: An interpretable stacked denoising auto-encoders for fan fault diagnosis on steelmaking workshops, Adv. Eng. Inform. 65 (2025) 103260.
[4] L. Long, J. Guo, H. Chu, S. Wang, S. Xu, L. Deng, Binocular vision-based pose monitoring technique for assembly alignment of precast concrete components, Adv. Eng. Inform. 65 (2025) 103205.
[5] D. Kumar, N.S. Ranawat, P.K. Kankar, A. Miglani, InceptionV3 based blockage fault diagnosis of centrifugal pump, Adv. Eng. Inform. 65 (2025) 103181.
[6] Y. Jiang, Z. Tan, J. Wang, X. Sun, M. Lin, H. Li, GiraffeDet: A heavy-neck paradigm for object detection, 2022, arXiv preprint arXiv:2202.04256.
[7] K.P. Alexandridis, J. Deng, A. Nguyen, S. Luo, Adaptive parametric activation, in: European Conference on Computer Vision, Springer, 2025, pp. 455–476.
[8] D. Hendrycks, K. Gimpel, Gaussian error linear units (gelus), 2016, arXiv preprint arXiv:1606.08415.
[9] Z. Zhou, W. Zhao, K. Song, Y. Wang, J. Li, EAFNet: Extraction-amplification-fusion network for tiny cracks detection, Eng. Appl. Artif. Intell. 134 (2024) 108691.
[10] C.C. Horuz, G. Kasenbacher, S. Higuchi, S. Kairat, J. Stoltz, M. Pesl, B.A. Moser, C. Linse, T. Martinetz, S. Otte, The resurrection of the ReLU, 2025, arXiv preprint arXiv:2505.22074.
[11] F. Cui, Q. Cui, Y. Song, A survey on learning-based approaches for modeling and classification of human–machine dialog systems, IEEE Trans. Neural Netw. Learn. Syst. 32 (4) (2020) 1418–1432.
[12] S.B. Block, R.D. da Silva, L.B. Dorini, R. Minetto, Inspection of imprint defects in stamped metal surfaces using deep learning and tracking, IEEE Trans. Ind. Electron. 68 (5) (2020) 4498–4507.
[13] L. Xie, X. Xiang, H. Xu, L. Wang, L. Lin, G. Yin, FFCNN: A deep neural network for surface defect detection of magnetic tile, IEEE Trans. Ind. Electron. 68 (4) (2020) 3506–3516.
[14] J. Sun, C. Li, X.-J. Wu, V. Palade, W. Fang, An effective method of weld defect detection and classification based on machine vision, IEEE Trans. Ind. Inform. 15 (12) (2019) 6322–6333.
[15] C. Hu, Y. Wang, An efficient convolutional neural network model based on object-level attention mechanism for casting defect detection on radiography images, IEEE Trans. Ind. Electron. 67 (12) (2020) 10922–10930.
[16] X. Zhou, Y. Wang, Q. Zhu, J. Mao, C. Xiao, X. Lu, H. Zhang, A surface defect detection framework for glass bottle bottom using visual attention model and wavelet transform, IEEE Trans. Ind. Inform. 16 (4) (2020) 2189–2201.
[17] Y. Zhao, Q. Liu, H. Su, J. Zhang, H. Ma, W. Zou, S. Liu, Attention-based multiscale feature fusion for efficient surface defect detection, IEEE Trans. Instrum. Meas. 73 (2024) 1–10.
[18] W. Luo, T. Niu, H. Yao, L. Tang, W. Yu, B. Li, Unsupervised defect segmentation via forgetting-inputting-based feature fusion and multiple hierarchical feature difference, IEEE Sens. J. 23 (13) (2023) 14448–14459.
[19] Y. Peng, F. Xia, C. Zhang, J. Mao, Deformation feature extraction and double attention feature pyramid network for bearing surface defects detection, IEEE Trans. Ind. Inform. (2024).
[20] S. Bai, L. Yang, Y. Liu, H. Yu, DMF-Net: A dual-encoding multi-scale fusion network for pavement crack detection, IEEE Trans. Intell. Transp. Syst. 25 (6) (2024) 5981–5996.

[21] A.L. Maas, A.Y. Hannun, A.Y. Ng, et al., Rectifier nonlinearities improve neural network acoustic models, in: Proceedings of the International Conference on Machine Learning, Vol. 30, Atlanta, GA, 2013, p. 3.
[22] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1026–1034.
[23] D.-A. Clevert, T. Unterthiner, S. Hochreiter, Fast and accurate deep network learning by exponential linear units (elus), 2015, arXiv preprint arXiv:1511.07289.
[24] G. Klambauer, T. Unterthiner, A. Mayr, S. Hochreiter, Self-normalizing neural networks, Adv. Neural Inf. Process. Syst. 30 (2017).
[25] S.R. Dubey, S. Chakraborty, Average biased ReLU based CNN descriptor for improved face retrieval, Multimedia Tools Appl. 80 (2021) 23181–23206.
[26] H. Li, J. Li, H. Wei, Z. Liu, Z. Zhan, Q. Ren, Slim-neck by GSConv: a lightweight-design for real-time detector architectures, J. Real-Time Image Process. 21 (3) (2024) 62.
[27] G. Jocher, YOLOv5 by Ultralytics, 2020, URL https://github.com/ultralytics/yolov5.
[28] G. Jocher, A. Chaurasia, J. Qiu, YOLO by Ultralytics, 2023, URL https://github.com/ultralytics/ultralytics.
[29] A. Wang, H. Chen, L. Liu, K. Chen, Z. Lin, J. Han, et al., YOLOv10: Real-time end-to-end object detection, Adv. Neural Inf. Process. Syst. 37 (2024) 107984–108011.
[30] Y. Tian, Q. Ye, D. Doermann, YOlOv12: Attention-centric real-time object detectors, 2025, arXiv preprint arXiv:2502.12524.
[31] S. Ren, K. He, R. Girshick, J. Sun, Faster RCNN: Towards real-time object detection with region proposal networks, 2016, arXiv preprint arXiv:1506.01497.
[32] C. Li, L. Li, H. Jiang, K. Weng, Y. Geng, L. Li, Z. Ke, Q. Li, M. Cheng, W. Nie, et al., YOLOv6: A single-stage object detection framework for industrial applications, 2022, arXiv preprint arXiv:2209.02976.
[33] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, L. Shao, Pyramid vision transformer: A versatile backbone for dense prediction without convolutions, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 568–578.
[34] C.-Y. Wang, A. Bochkovskiy, H.-Y.M. Liao, YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 7464–7475.
[35] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2980–2988.
[36] Z. Cai, N. Vasconcelos, Cascade R-CNN: Delving into high quality object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 6154–6162.
[37] Y. Zhao, W. Lv, S. Xu, J. Wei, G. Wang, Q. Dang, Y. Liu, 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. 16965–16974.
[38] C.-Y. Wang, I.-H. Yeh, 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.
[39] Z. Tian, C. Shen, H. Chen, T. He, FCOS: Fully convolutional one-stage object detection, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 9627–9636.
[40] M. Tan, R. Pang, Q.V. Le, EfficientDet: Scalable and efficient object detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 10781–10790.
[41] Y. Peng, H. Li, P. Wu, Y. Zhang, X. Sun, F. Wu, D-FINE: Redefine regression task in DETRs as fine-grained distribution refinement, 2024, arXiv preprint arXiv:2410.13842.
[42] Y. Feng, J. Huang, S. Du, S. Ying, J.-H. Yong, Y. Li, G. Ding, R. Ji, Y. Gao, Hyper-YOLO: When visual object detection meets hypergraph computation, IEEE Trans. Pattern Anal. Mach. Intell. 47 (4) (2025) 2388–2401.
[43] Z. Yu, Q. Guan, J. Yang, Z. Yang, Q. Zhou, Y. Chen, F. Chen, LSM-YOLO: A compact and effective roi detector for medical detection, in: International Conference on Neural Information Processing, 2025, pp. 30–44.
[44] Y. Xiao, T. Xu, Y. Xin, J. Li, FBRT-YOLO: Faster and better for real-time aerial image detection, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 39, 2025, pp. 8673–8681.
[45] Z. Yang, Q. Guan, Z. Yu, X. Xu, H. Long, S. Lian, H. Hu, Y. Tang, MHAF-YOLO: Multi-branch heterogeneous auxiliary fusion YOLO for accurate object detection, 2025, arXiv preprint arXiv:2502.04656.
[46] M. Yang, H. Bai, J. Hu, D. Li, A dynamic context-aware aggregation strategy for small object detection, Pattern Recognit. (2025) 112127.
[47] J. Yang, S. Liu, J. Wu, X. Su, N. Hai, X. Huang, Pinwheel-shaped convolution and scale-based dynamic loss for infrared small target detection, 2024, arXiv preprint arXiv:2412.16986.
[48] B. Zhao, X. Zhou, G. Yang, J. Wen, J. Zhang, J. Dou, G. Li, X. Chen, B.M. Chen, High-resolution infrastructure defect detection dataset sourced by unmanned systems and validated with deep learning, Autom. Constr. 163 (2024) 105405.
[49] H. Han, R. Yang, S. Li, R. Hu, X. Li, SSGD: A smartphone screen glass dataset for defect detection, in: ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2023, pp. 1–5.
[50] Y. He, K. Song, Q. Meng, Y. Yan, An end-to-end steel surface defect detection approach via fusing multiple hierarchical features, IEEE Trans. Instrum. Meas. 69 (4) (2019) 1493–1504.
[51] S. Zhang, C. Chi, Y. Yao, Z. Lei, 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, 2020, pp. 9759–9768.
[52] X. Li, W. Wang, L. Wu, S. Chen, X. Hu, J. Li, J. Tang, J. Yang, Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection, Adv. Neural Inf. Process. Syst. 33 (2020) 21002–21012.
[53] Z. Ge, S. Liu, F. Wang, Z. Li, J. Sun, YOLOX: Exceeding yolo series in 2021, 2021, arXiv preprint arXiv:2107.08430.
[54] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin TransFormer: Hierarchical vision transformer using shifted windows, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 10012–10022.
[55] R. Yang, H. Ma, J. Wu, Y. Tang, X. Xiao, M. Zheng, X. Li, ScalableViT: Rethinking the context-oriented generalization of vision transformer, in: European Conference on Computer Vision, Springer, 2022, pp. 480–496.
[56] K. Li, Y. Wang, P. Gao, G. Song, Y. Liu, H. Li, Y. Qiao, Uniformer: Unified transformer for efficient spatiotemporal representation learning, 2022, arXiv preprint arXiv:2201.04676.
[57] I. El Jaafari, A. Ellahyani, S. Charfi, Parametric rectified nonlinear unit (PRenu) for convolution neural networks, Signal Image Video Process. 15 (2) (2021) 241–246.
[58] J.T. Barron, Continuously differentiable exponential linear units, 2017, arXiv preprint arXiv:1704.07483.
[59] G.K. Pandey, S. Srivastava, ResNet-18 comparative analysis of various activation functions for image classification, in: 2023 International Conference on Inventive Computation Technologies, IEEE, 2023, pp. 595–601.