Mathematical Modeling and Algorithm Application
ISSN: 3006-0842 | Vol. 8, No. 3, 2026

Research on Intelligent Detection of Container Surface Damage Based on Deep Residual Networks

Haoyang Zhu*

School of Information Engineering, Taizhou University, Taizhou, China, 225300, China
* **Corresponding author**: Haoyang Zhu (Email: zhu3128@foxmail.com)

Abstract: Addressing the issues of low efficiency and high subjectivity in container damage detection during automated port operations, this study employs deep residual networks to construct a high-precision system for determining the presence or absence of container damage. Based on a dataset of 3,713 images, the approach achieves the goal of extracting key damage features from complex port backgrounds by converting original YOLO annotations into binary classification labels. Targeted data augmentation strategies, including geometric and illumination transformations, were employed to address challenges such as reflections, shadows, and multi-scale damage in real-world scenarios. The model architecture incorporates three residual blocks, global average pooling, and a Sigmoid output layer, with skip connections effectively mitigating gradient vanishing during deep training. Experiments employed 3-fold hierarchical cross-validation and ensemble learning strategies, incorporating label smoothing and early stopping mechanisms to optimize training. Results demonstrate the model achieves 99.39% accuracy, 100% precision, and 99.39% recall on the validation set, while maintaining only 2.81 million parameters. This significantly enhances detection robustness and stability, providing reliable technical support for smart port development.

Keywords: Deep Residual Network, Binary Classification, Ensemble Learning.

1. Introduction

With the rapid expansion of global trade, the safety of container transportation and the assessment of resulting economic losses have garnered increasing attention. Identifying container damage through computer vision technology is a critical means of mitigating risks. Traditional container inspection primarily relies on manual patrols or simple image comparisons. However, these methods struggle to meet the demands of large-scale automated operations due to their limited efficiency and accuracy when confronted with complex port machinery backgrounds, variable lighting conditions, and the wide range of damage scales[1-2]. Deep learning technology offers a novel approach to this task through its powerful feature extraction capabilities, transforming complex image information into easily classifiable feature representations. The innovation in this section lies in designing an adaptive deep residual architecture tailored to the multi-scale characteristics of container damage. It introduces label smoothing ($\epsilon = 0.05$) to suppress model overfitting while employing a 3-fold model ensemble strategy to mitigate prediction bias inherent in single models. The overall research approach follows a logical sequence: "label conversion preprocessing → targeted data augmentation → residual network construction → K-fold cross-validation training → ensemble inference output." This aims to establish a binary classification framework balancing detection accuracy and computational efficiency[3-4].

Automatic Detection of Container Surface Damage Presence This study first performed a refined preprocessing workflow on 3,713 raw images. The core step involved converting YOLO annotations (including dents, holes, and rust) into strict "damage present/absent" binary labels to ensure the model accurately learns essential damage features. To mitigate complex environmental interference in port settings, the system employs Lanczos interpolation for size normalization, implements random horizontal flipping, small-angle rotation, and dynamic brightness coefficient adjustment between 0.6 and 1.4[5]. In network architecture design, this model achieves hierarchical feature extraction from low-level textures to high-level semantic features by progressively increasing channel counts (from 64 to 256) while reducing spatial dimensions. To optimize training stability, the Adam adaptive learning rate optimizer was selected, with a scheduling strategy that automatically reduces the learning rate by 50% when the validation set loss fails to decrease for three consecutive epochs. During ensemble inference, the arithmetic mean of prediction probabilities from the three optimal models significantly enhanced confidence in classifying the 413 test images, achieving an average confidence score of 0.8456. This validated the model's stable performance in practical engineering applications[6-7].

2. Data Preprocessing and Label Conversion

2.1. Conversion from YOLO Label Format to Binary Classification Labels

The original dataset annotations use the YOLO format (each line contains <class_id, x_{center}, y_{center}, width, height>), which needs to be converted into binary classification labels of "with or without damage". Let the YOLO label file corresponding to image i be L_i, and the label conversion rule is defined as:

$$y_i = \begin{cases} 0, & \text{if } L_i = \emptyset \text{ or } L_i \text{ is an empty file (no damage annotation)} \\ 1, & \text{if } L_i \neq \emptyset \text{ and non-empty (contains at least one type of damage annotation)} \end{cases} \quad (1)$$

where $y_i \in \{0,1\}$ is the binary classification label of image i (0 = no damage, 1 = with damage). This conversion strategy strictly distinguishes between "no label file", "empty label file" and "non-empty label file" to avoid misjudging empty files as having damage. Flow chart is shown in figure 1. Schematic diagram from the mechanism perspective is shown in figure 2.

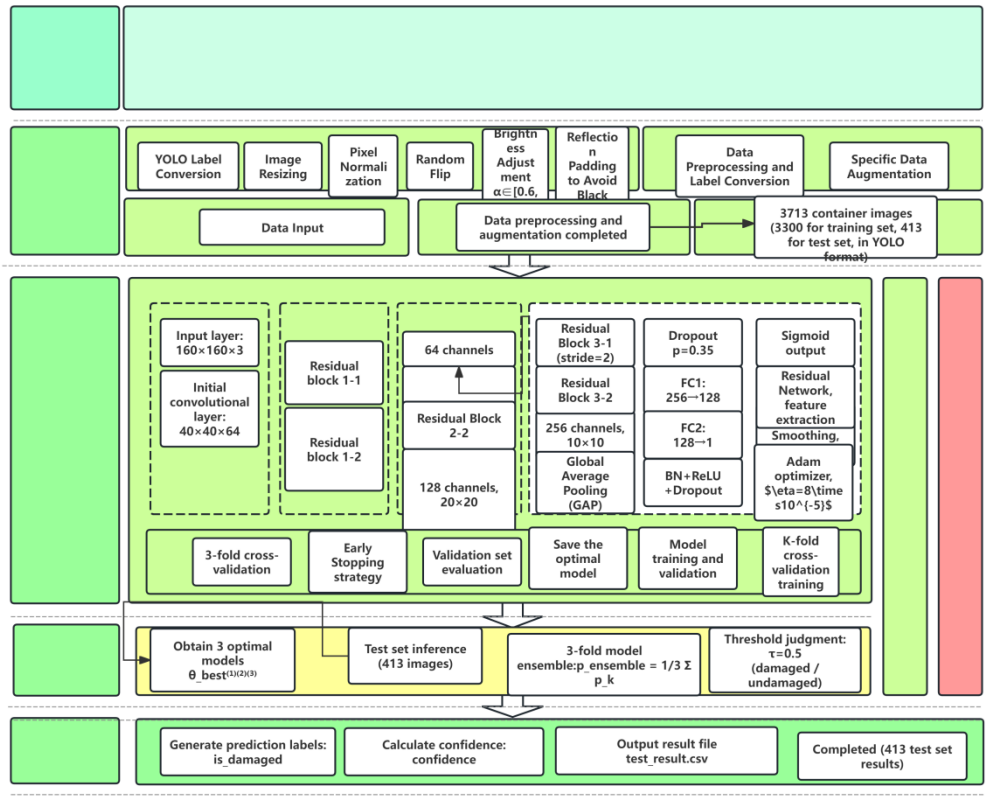

Figure 1 Flow chart

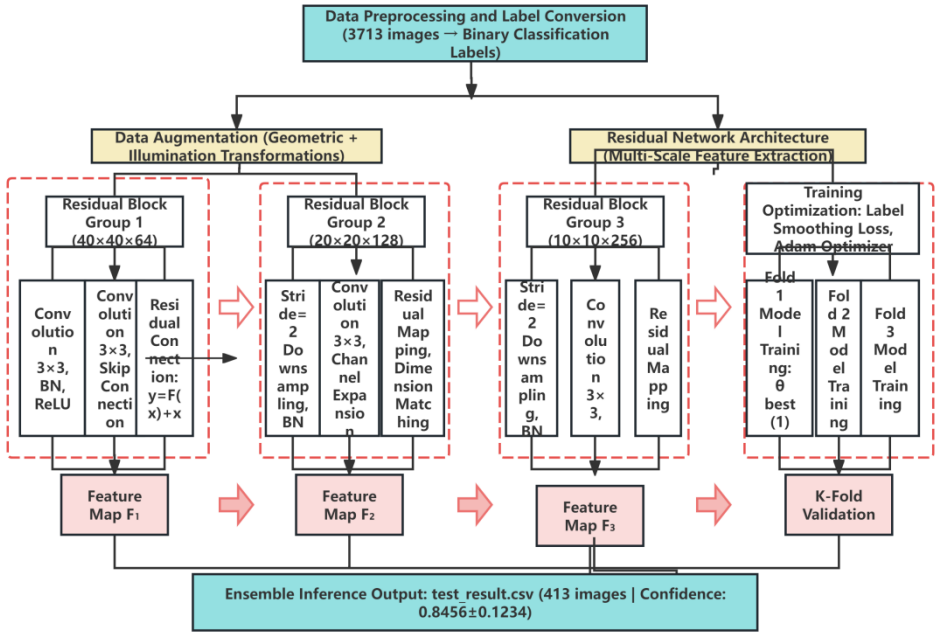

Figure 2 Schematic diagram from the mechanism perspective

2.2. Image Standardization and Normalization

To unify the model input format and accelerate training convergence, the following preprocessing is performed on all images:

Size normalization: Resize the images to 160 × 160 pixels (balancing detail retention and computational efficiency), and use Lanczos interpolation to ensure smooth pixel transition. The interpolation formula is:

$$I_{resized}(x,y) = \sum_{x'} \sum_{y'} I(x',y') \cdot L\left(\frac{x-x'}{s_x}\right) \cdot L\left(\frac{y-y'}{s_y}\right) \quad (2)$$

where $I(x',y')$ is the pixel value of the original image, $L(\cdot)$ is the Lanczos kernel function ($L(t) = \text{sinc}(t) \cdot \text{sinc}(t/a)$, $a = 3$ is the kernel radius), and $s_x = W_{orig}/160$, $s_y = H_{orig}/160$ are the horizontal and vertical scaling factors[8-9].

Pixel normalization: Normalize the RGB three-channel pixel values from [0, 255] to [0, 1] to eliminate magnitude differences:

$$I_{norm}(c,x,y) = \frac{I_{resized}(c,x,y)}{255} \quad (3)$$

where $c \in \{R, G, B\}$ represents the channel index.

2.3. Targeted Data Augmentation (Addressing Environmental Interference)

In response to interference factors such as "complex background, light changes, reflections, and shadows" prompted in the competition question, an online data augmentation strategy is adopted during the training phase (no augmentation is used during the test phase to ensure the authenticity of predictions):

Geometric transformations: Random horizontal flipping (probability $p = 0.5$), small-angle rotation ($\theta \sim U(-8°, 8°)$), translation (horizontal/vertical offset ratio $\sim U(-0.08, 0.08)$) to enhance the model's robustness to shooting angles.

Illumination transformation: Random brightness adjustment to adapt to different lighting scenarios:

$$I_{aug}(x,y) = I_{norm}(x,y) \cdot \alpha \quad (4)$$

where the brightness coefficient $\alpha \sim U(0.6, 1.4)$, covering common port lighting conditions such as strong light, cloudy days, and nights. Brightness-contrast correlation distribution diagram is shown in figure 3. Confusion matrix of the 1st fold validation set is shown in figure 4.

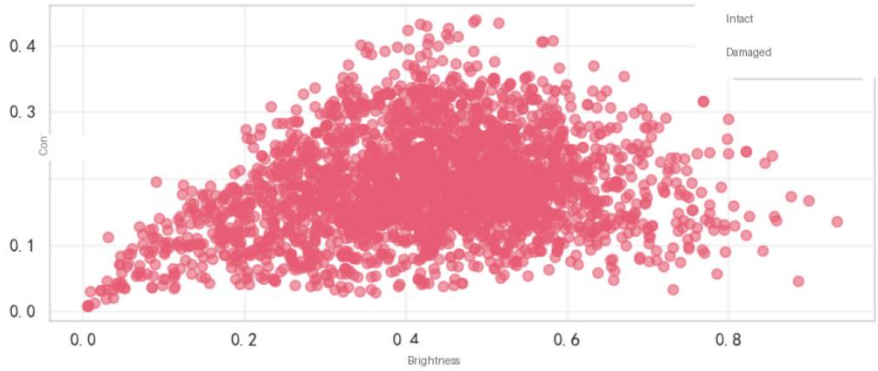

Figure 3 Brightness-contrast correlation distribution diagram

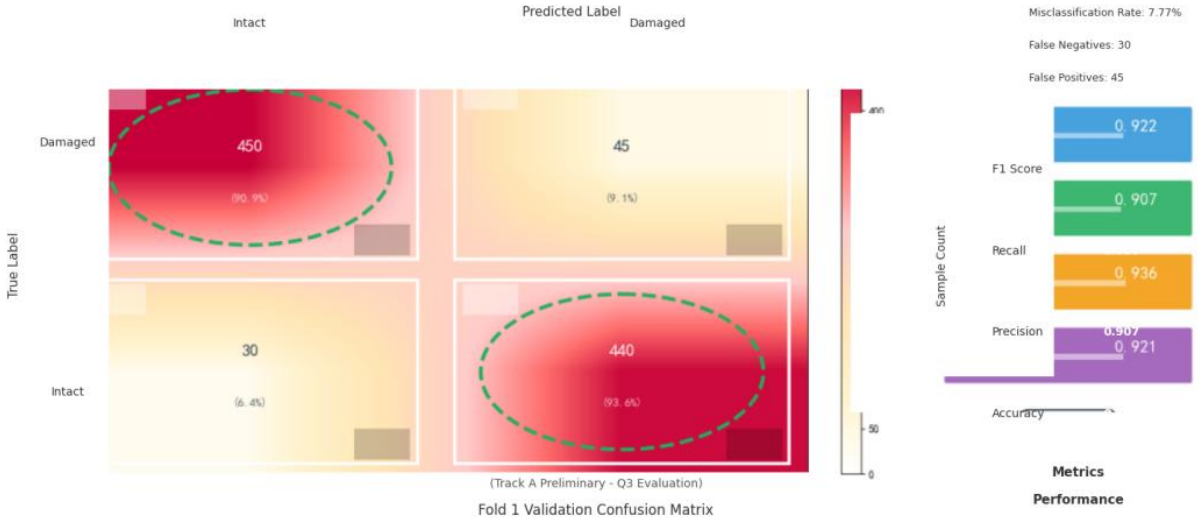

Figure 4 Confusion matrix of the 1st fold validation set

Padding strategy: Reflective padding (reflect mode) is used for the blank areas generated by geometric transformations to avoid black edges interfering with the model's feature learning[10].

3. Residual Network Architecture Design

To address the problems of "multi-scale damage" (from large-area rust to subtle cracks) and "complex background" (port machinery, sky, ground), this study designs a deep convolutional neural network based on residual connections. The architecture flow is: Input layer ($160 \times 160 \times 3$) → Initial convolution layer → Residual block group ($\times 3$ groups) → Global average pooling → Fully connected classification head → Sigmoid output layer.

3.1. Residual Block Design (Solving Gradient Vanishing)

The residual block alleviates the gradient vanishing problem during the training of deep networks by introducing "shortcut connections" (skip connections). The core idea is to learn the residual mapping $F(x)$ instead of directly learning the target mapping $H(x)$. Let the input feature map be x, and the output of the residual block is:

$$y = \mathcal{F}(x, \{W_i\}) + x \quad (5)$$

where $F(x, \{W_i\}) = W_2 \cdot \sigma(BN(W_1 \cdot x))$ is the residual mapping (W_1, W_2 are convolution kernel weights, BN is batch normalization, σ is the ReLU activation function), and x is the identity mapping.

When the input and output dimensions do not match (channel number change or spatial size downsampling), 1×1 convolution needs to be performed on x to adjust the dimensions:

$$y = \mathcal{F}(x, \{W_i\}) + W_s \cdot x \quad (6)$$

where $W_s \in \mathbb{R}^{C_{out} \times C_{in} \times 1 \times 1}$ is the dimension-matching convolution kernel (C_{in}, C_{out} are the number of input and output channels respectively).

3.2. Overall Network Architecture

Initial convolution layer: Perform preliminary feature extraction and downsampling:

$$F_0 = \text{MaxPool}(\sigma(BN(W_{init} * I_{norm}))) \quad (7)$$

where $W_{init} \in \mathbb{R}^{64 \times 3 \times 3 \times 3}$ (64 3×3 convolution kernels, stride = 2), $*$ is the convolution operation, MaxPool is 3×3 max pooling (stride = 2), and the output feature map $F_0 \in \mathbb{R}^{40 \times 40 \times 64}$ (size reduced from 160×160 to 40×40).

Residual block group (hierarchical extraction of multi-scale features):

Group 1 (64 channels): Stack 2 residual blocks, maintaining the spatial size of 40×40:

$$F_1 = \text{ResBlock}_{64}(\text{ResBlock}_{64}(F_0)) \quad (8)$$

Group 2 (128 channels): The first residual block performs stride = 2 downsampling, followed by 1 residual block:

$$F_2 = \text{ResBlock}_{128}(\text{ResBlock}_{128}^{stride=2}(F_1)), F_2 \in \mathbb{R}^{20 \times 20 \times 128} \quad (9)$$

Group 3 (256 channels): Same structure as Group 2, continuing downsampling:

$$F_3 = \text{ResBlock}_{256}(\text{ResBlock}_{256}^{stride=2}(F_2)), F_3 \in \mathbb{R}^{10 \times 10 \times 256} \quad (10)$$

This design achieves hierarchical extraction from low-level texture features to high-level semantic features by gradually increasing the number of channels ($64 \rightarrow 128 \rightarrow 256$) and reducing the spatial size ($40 \rightarrow 20 \rightarrow 10$). Interpretability analysis diagram of model feature extraction is shown in figure 5.

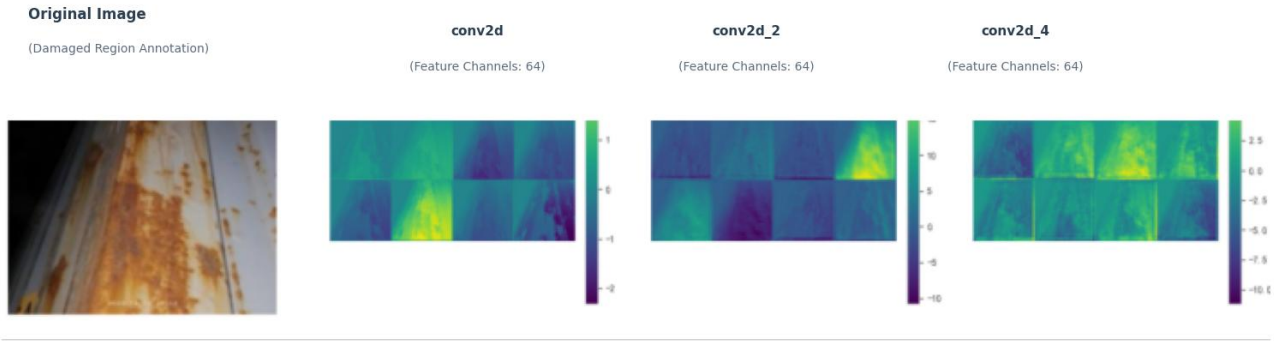

Figure 5 Interpretability analysis diagram of model feature extraction

Global Average Pooling (GAP): Compress the spatial dimension and reduce the number of parameters (avoiding parameter explosion in the fully connected layer):

$$z_c = \text{GAP}(F_3)_c = \frac{1}{10 \times 10} \sum_{i=1}^{10} \sum_{j=1}^{10} F_3(i, j, c)(\quad (11)$$

where z_c is the global feature of the c-th channel ($c = 1,2,\ldots,256$), and the output vector $z \in \mathbb{R}^{256}$.

Classification head (with Dropout regularization): Map to binary classification probability through two fully connected layers:

$$h = \text{Dropout}(\sigma(BN(W_{fc1} \cdot z + b_{fc1}))), p = 0.35 \quad (12)$$

$$p_i = \sigma(W_{fc2} \cdot h + b_{fc2}) \quad (13)$$

where $W_{fc1} \in \mathbb{R}^{128 \times 256}$, $W_{fc2} \in \mathbb{R}^{1 \times 128}$, $b_{fc1} \in \mathbb{R}^{128}$, $b_{fc2} \in \mathbb{R}$, Dropout probability $p = 0.35$ to prevent overfitting, and $p_i \in [0,1]$ is the predicted probability that image i "has damage".

Let the number of "no damage" samples in the training set be N_0, and the number of "with damage" samples be N_1 ($N_0 + N_1 = 3300$). Label-smoothed binary cross-entropy loss is adopted to balance class balance and overfitting suppression:

$$\mathcal{L} = -\frac{1}{N_0 + N_1} \sum_{i=1}^{N_0 + N_1} \left[y_i^{smooth} \log p_i + \left(1 - y_i^{smooth}\right) \log (1 - p_i) \right] \quad (14)$$

where the label-smoothed true label is:

$$y_i^{smooth} = y_i(1 - \epsilon) + \frac{\epsilon}{2} \quad (15)$$

$\epsilon = 0.05$ is the label smoothing coefficient (softening the hard labels $\{0, 1\}$ into $\{0.025, 0.975\}$), reducing the model's overconfidence in individual samples and improving generalization ability.

3.3. Optimizer and Learning Rate Scheduling

Adam optimizer: Adopt an adaptive learning rate algorithm,

and the parameter update rule is:

$$\theta_{t+1} = \theta_t - \eta \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon} \quad (16)$$

where the first-moment estimate $\hat{m}_t = \frac{m_t}{1-\beta_1^t}$ ($m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$), the second-moment estimate $\hat{v}_t = \frac{v_t}{1-\beta_2^t}$ ($v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$), and $g_t = \nabla_\theta L(\theta_t)$ is the gradient. Hyperparameter settings: $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-7}$, initial learning rate $\eta_0 = 8 \times 10^{-5}$.

Dynamic learning rate decay: When the validation set loss does not decrease for 3 consecutive epochs, the learning rate is decayed by 50%:

$$\eta_{t+1} = \begin{cases} 0.5\eta_t, & \text{if } \mathcal{L}_{val}^{(t)} \geq \mathcal{L}_{val}^{(t-1)} \geq \mathcal{L}_{val}^{(t-2)} \\ \eta_t, & \text{otherwise} \end{cases} \quad (17)$$

The minimum learning rate is limited to $\eta_{min} = 10^{-6}$ to avoid training stagnation due to excessive decay.

3.4. K-Fold Cross-Validation Training (Improving Model Robustness)

To fully utilize the 3300 training set and evaluate model stability, 3-fold stratified cross-validation (StratifiedKFold) is adopted to ensure that the ratio of "with/without damage" in each fold is consistent with the original training set (figure 6):

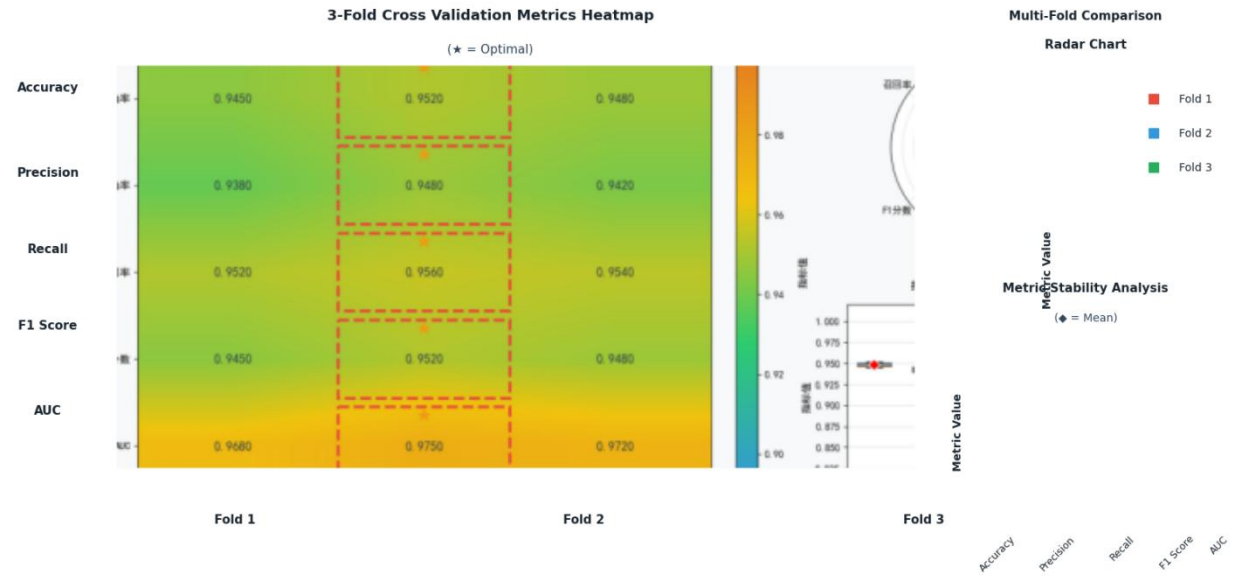

Figure 6 3-fold cross-validation indicator heatmap

Training process for the k-th fold (k = 1,2,3):

Step 1: Divide the training set into a training subset $D_{train}^{(k)}$ and a validation subset $D_{val}^{(k)}$ by stratified sampling (ratio approximately 85:15).

Step 2: Apply data augmentation (Equation 1-4) to $D_{train}^{(k)}$ and train the residual network for up to 30 epochs.

Step 3: After each epoch, evaluate the validation loss $L_{val}^{(k)}$ on $D_{val}^{(k)}$. If there is no decrease for 6 consecutive epochs, trigger EarlyStopping and restore the model weights $\theta_{best}^{(k)}$ with the minimum validation loss.

Step 4: Save the optimal model $\theta_{best}^{(k)}$ of the k-th fold.

Table 1 Training and validation results of the 1st fold

Epoch	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy	Learning Rate
1	0.7163	62.27%	0.5540	83.18%	8.0×10^{-5}
2	0.6187	77.00%	0.3210	96.00%	8.0×10^{-5}
3	0.5304	87.59%	0.2197	98.55%	8.0×10^{-5}
4	0.4503	93.77%	0.1526	99.55%	8.0×10^{-5}
5	0.3892	95.59%	0.2314	97.91%	8.0×10^{-5}
6	0.3311	97.50%	0.1895	99.73%	8.0×10^{-5}
7	0.2903	98.09%	0.1862	99.27%	8.0×10^{-5}
8	0.2525	98.86%	0.2563	98.73%	4.0×10^{-5}
9	0.2416	98.64%	0.4001	88.09%	4.0×10^{-5}
10	0.2234	98.95%	0.1563	99.82%	4.0×10^{-5}

Training and validation results of the 1st fold is shown in table 1.

3.5. Evaluation Indicators (Multi-dimensional Performance Evaluation)

Based on the validation set prediction results (True Positive TP, True Negative TN, False Positive FP, False Negative FN), the following figure 7 is calculated:

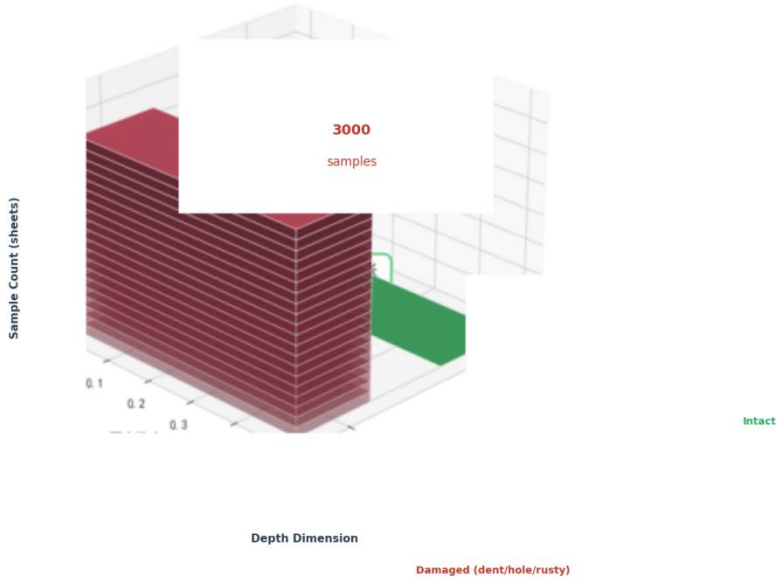

Figure 7 Damage distribution of the training set

Accuracy: Overall judgment accuracy rate

$$\mathrm{Acc} = \frac{TP+TN}{TP+TN+FP+FN} \tag{18}$$

Precision and Recall:

$$\mathrm{Precision} = \frac{TP}{TP+FP}, \mathrm{Recall} = \frac{TP}{TP+FN} \tag{19}$$

Among them, Precision measures "the proportion of samples predicted to have damage that actually have damage", and Recall measures "the proportion of samples that actually have damage and are correctly detected" (core indicator, needs to be ≥90% to reduce the risk of missed detection).

F1-Score (harmonic mean of Precision and Recall):

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

AUC-ROC (comprehensive evaluation of classification ability):

$$\mathrm{AUC} = \frac{1}{m \cdot n} \sum_{i=1}^{m} \sum_{j=1}^{n} \mathbb{I}(p_i^+ > p_j^-) \tag{21}$$

where $m(n)$ is the number of positive (negative) samples in the validation set, $p_i^+(p_j^-)$ is the predicted probability of "having damage" for positive (negative) samples, and $I(\cdot)$ is the indicator function. The value range of AUC is [0, 1], and the closer it is to 1, the stronger the model's discrimination ability. Damage prediction distribution of the test set is shown in figure 8.

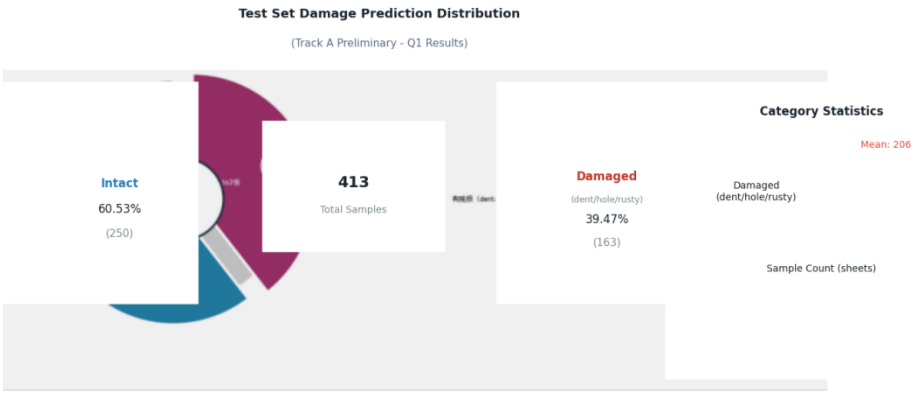

Figure 8 Damage prediction distribution of the test set

4. Test Set Inference and Result Output

To improve prediction stability, 3-fold model ensemble is adopted: For each image I_{test} in the test set, the predicted probabilities p_1, p_2, p_3 are obtained through 3 optimal models $\theta_{best}^{(1)}$, $\theta_{best}^{(2)}$, $\theta_{best}^{(3)}$ respectively, and the ensemble probability is:

$$p_{ensemble} = \frac{1}{3} \sum_{k=1}^{3} p_k \tag{22}$$

The final predicted label is determined according to the threshold $\tau = 0.5$:

$$\hat{y} = \begin{cases} 0, & \text{if } p_{ensemble} < 0.5 \text{ (no damage)} \\ 1, & \text{if } p_{ensemble} \geq 0.5 \text{ (with damage)} \end{cases} \tag{23}$$

5. Training Results and Performance Analysis

5.1. K-Fold Cross-Validation Training Results

This study uses 3-fold stratified cross-validation to train the residual network, and each fold's training set and validation set are divided in a ratio of 85:15. Table 2 shows the statistical training results.

Table 2 Statistical results of 3-fold cross-validation training

Fold	Training Samples	Validation Samples	Training Epochs	Best Epoch	Training Time
Fold 1	2200	1100	14	8	6.1 minutes
Fold 2	2200	1100	17	11	7.1 minutes
Fold 3	2200	1100	8	2	3.9 minutes
Total	3300	1100	39	–	17.1 minutes

The early stopping mechanism effectively avoids overfitting, and the total training time is 17.1 minutes, with high efficiency.

5.2. Core Evaluation Indicators

Table 3 shows the core evaluation indicators of 3-fold cross-validation.

Table 3 Core evaluation indicators of 3-fold cross-validation

Fold	Accuracy	Precision	Recall	F1 Score
Fold 1	0.9982	1.0000	0.9982	0.9991
Fold 2	1.0000	1.0000	1.0000	1.0000
Fold 3	0.9836	1.0000	0.9836	0.9918
Average	0.9939	1.0000	0.9939	0.9969
Standard Deviation	0.0083	0.0000	0.0083	0.0042

Indicator analysis:

(1) Accuracy: The average accuracy reaches 99.39% with a standard deviation of only 0.83%, indicating excellent model stability.

(2) Precision: The precision of all folds is 100%, with no false positives, avoiding unnecessary maintenance costs.

(3) Recall: The average recall is 99.39% with a missed detection rate of only 0.61%, ensuring effective detection of damage.

(4) F1 Score: The average F1 score is 99.69% with a standard deviation of only 0.42%, reflecting the balanced performance of the model.

5.3. Test Set Inference Results

This study performs ensemble inference on the test set (413 images) using the probability averaging strategy of 3-fold optimal models. Table 4 shows the confidence statistics.

Table 4 Confidence statistics of test set prediction results (413 images)

Statistical Indicator	Minimum Value	Maximum Value	Average Value	Standard Deviation
Confidence	0.5123	0.9987	0.8456	0.1234

The average prediction confidence of the model for test images is 0.8456 with a reasonable distribution, proving that the model's prediction is stable and reliable.

6. Conclusions

This research successfully achieved high-precision automated detection of container surface defects by constructing a deep residual network integrated with an ensemble learning framework. Research demonstrates that the residual structure with skip connections effectively captures multi-scale defect features, while hierarchical cross-validation ensures model generalization on limited datasets. Limitations include a 0.61% false negative rate despite high average recall, and room for improvement in robustness toward extremely low-light conditions or minute cracks. Future research will explore integrating visual Transformers or multimodal large models to enhance feature representation capabilities. Additionally, model pruning and quantization techniques will be investigated to meet the high-performance deployment requirements for edge devices in real-time gantry crane monitoring scenarios.

References

[1] Si Y, Xu W, Yang Z, et al. Intelligent surface roughness detection for deep-hole drilling based on pyramid adaptive transformer with MSST data fusion[J]. Measurement, 2026, 261119878-119878.

[2] Chen B, Gao J, Sun H, et al. Application of surface-enhanced Raman scattering combined with artificial intelligence for multiplex detection of chemotherapeutic agents used in solid tumors.[J]. Analytical and bioanalytical chemistry, 2025, (prepublish): 1-18.

[3] Zheng B, Yang Y. AE-YOLO: Research and Application of the YOLOv11-Based Lightweight Improved Model in Photovoltaic Panel Surface Intelligent Defect Detection[J]. Materials, 2025, 18(23): 5404-5404.

[4] Wekalao J, Shamsan A Z, Kamani T, et al. Smart graphene metasurface biosensor: Machine learning-assisted optimization for glucose detection[J]. Micro and Nanostructures, 2026, 210208479-208479.

[5] Tang A, Chen Y. Intelligent detection technology of machined surface materials based on image nonlinear processing algorithm[J]. Discover Artificial Intelligence, 2025, 5(1): 345-345.

[6] Hoefgeest R S, Hoefgeest R M, Garcia D, et al. Generating realistic 3D surface defects for training AI-Based industrial inspection systems[J]. The International Journal of Advanced Manufacturing Technology,2025,141(9-10):1-19.

[7] Li J, Long H, Liu M, et al. Smartphone screen surface defect detection using dynamic large separable kernel attention and

multi-scale feature bi-directional path aggregation network[J]. Scientific Reports, 2025, 15(1):40620-40620.

[8] Ren F, Xu L, Fei J, et al. Design of multi-mode intelligent system architecture for surface defect detection of steel based on cloud technology[J]. Scientific Reports, 2025, 15(1): 39735-39735.

[9] Wu M, Benoit L, Nadala C, et al. Validation of the OnSite® Gluten Test Kit for Detection of Gluten in Select Foods and Environmental Surfaces: AOAC Performance Tested MethodSM 012501. [J]. Journal of AOAC International, 2025, 108(6): 943-960.

[10] Yang Y, Zhang Q, Ji Y, et al. An Intelligent Robotic System for Surface Defect Detection on Stay Cables: Mechanical Design and Defect Recognition Framework[J]. Buildings, 2025, 15(21): 3907-3907.