Theory and Practice of Science and Technology
2025, VOL. 6, NO. 10, 72-80
DOI: 10.47297/taposatWSP2633-456914.20250610

Optimization of Mobile Screen Defect Detection Algorithm Based on Lightweight YOLOv8

Hanlin Huang

School of Advanced Manufacturing and School of Marine Sciences, Fuzhou University, Quanzhou, Fujian, 362200, China

ABSTRACT
With the rapid advancement of smart manufacturing and the consumer electronics industry, the surface quality of smartphone screens has become a critical indicator for evaluating product reliability and user experience. Traditional manual inspection methods struggle to meet the demands for efficiency and consistency in modern production. Consequently, automated visual inspection based on deep learning has emerged as a key development direction for industrial quality control. Addressing the limitations of existing YOLOv8 models in detecting small objects and high computational complexity, this paper proposes an improved algorithm, YOLOv8s-M4SE, which integrates a lightweight backbone network with a channel attention mechanism. Using a publicly available mobile screen defect dataset, we expand the sample distribution through multi-dimensional data augmentation strategies. In terms of network architecture, the original YOLOv8s backbone is replaced with MobileNetV4ConvSmall to reduce computational complexity, while introducing the SE channel attention module to enhance defect feature representation. Experimental results demonstrate that the YOLOv8s-M4SE model achieves an excellent balance between detection accuracy and computational efficiency. It provides a viable solution for applying deep learning in quality inspection for consumer electronics manufacturing.

KEYWORDS
YOLOv8; MobileNetV4; SE channel attention Module; Smartphone screen defect detection

1 Introduction

With the rapid advancement of smart manufacturing and the consumer electronics industry, the surface quality of smartphone screens has become a critical indicator for evaluating product reliability and user experience. During production, screens are highly susceptible to minor defects such as oil, scratches, and stains. These defects typically feature small areas, low contrast, blurred edges, and variable shapes, placing higher demands on the resolution capability and robustness of automated detection algorithms. Traditional manual inspection methods are not only inefficient and highly subjective but also struggle to meet the high-speed cadence and stringent consistency demands of modern production lines. Consequently, deep learning-based visual inspection technology has emerged as the primary direction for industrial quality control automation.

In recent years, object detection algorithms represented by convolutional neural networks (CNNs) have achieved significant results in both general object detection and industrial surface defect recognition [1]. However, directly applying these models to smartphone screen surface defect detection presents several challenges: On one hand, screen defects often exhibit characteristics such as high light reflection, weak texture, and low contrast, making it difficult for networks to effectively distinguish defects from backgrounds during feature extraction. On the other hand, while mainstream models like YOLOv8 offer high detection accuracy, their complex network structures and large parameter counts limit deployment efficiency in embedded or real-time detection scenarios.

To address these issues, researchers have attempted to enhance detection performance or reduce computational complexity by modifying network architectures. For instance, Yang et al [2]. proposed the Blade-YOLOv8 network based on YOLOv8 for wind turbine blade defect detection; Wu et al [3]. introduced the SSB-YOLOv8 model based on YOLOv8 for lump coal detection on mine conveyor belts; Liu et al [4]. proposed the SFC-YOLOv8 model to enhance detection accuracy and robustness for steel strip surface defects. However, similar research has primarily focused on industries like steel, metal, and coal mining, with limited studies addressing defect detection on smartphone screens. Therefore, constructing an efficient detection network tailored for smartphone screen surface defects, while balancing lightweight design and accuracy, holds significant engineering application value.

Based on this, this paper proposes an improved lightweight and attention-fused model, YOLOv8s-M4SE, using YOLOv8s as the baseline to address its shortcomings in small object detection and high computational complexity. Specific improvements include: (1) designing a multidimensional data augmentation strategy to enhance model robustness and generalization capability; (2) replacing the backbone network with MobileNetV4ConvSmall to reduce model complexity; (3) Embedding a SE channel attention module to strengthen responses to critical defect channels. Through these designs, the model achieves a highly efficient detection framework by substantially reducing computational load while maintaining detection accuracy to the greatest extent possible.

The structure of this paper is as follows: Chapter 2 introduces the dataset used in the experiments, including its source,

annotation format conversion, and manual data augmentation methods; Chapter 3 elaborates on the fundamental principles and network architecture of the YOLOv8 algorithm; Chapter 4 describes the design of the improved YOLOv8s-M4SE model; Chapter 5 presents the experimental environment, evaluation metrics, and comparative analysis of experimental results to validate the effectiveness of the proposed model; Chapter 6 summarizes the main research contributions and innovations of this paper and outlines future research directions.

2 Dataset Preparation

2.1 Data Source and Conversion

The image data used in this experiment originates from the Smartphone Screen Surface Defect Segmentation Dataset released by the Peking University Intelligent Robotics Open Laboratory. This dataset is specifically constructed for industrial quality inspection visual tasks, simulating the lighting conditions and environments encountered on production lines for smartphone screens in real manufacturing settings.

The dataset contains three common screen defect categories:

(1) Oil: Surface oil contamination with irregular distribution;

(2) Scratch: Long, thin defects with sharp edges;

(3) Stain: Localized bright or dark areas with blurred boundaries.

Each defect category comprises 400 images, totaling 1,200 images. All images were captured using industrial cameras under standardized lighting conditions, ensuring high-quality, high-resolution characteristics.

The original dataset was annotated in PASCAL VOC format. To adapt to the YOLOv8 model training workflow, the data underwent format conversion for experimentation. The dataset was divided into training, validation, and test sets at a ratio of 6:2:2. This dataset features clearly defined defect types, high image quality, and standardized annotation accuracy, making it suitable for the research tasks in this paper.

2.2 Data Augmentation Methods

To enhance the model's robustness and generalization capabilities across varying defect morphologies, lighting conditions, and shooting angles, multiple data augmentation strategies were designed and implemented based on the original dataset. These augmentations were implemented using the Albumentations image augmentation library [5], encompassing spatial transformations, color perturbations, and blurring/sharpening techniques.

Key augmentation operations include:

(1) Blurring (GaussianBlur): Simulates lens blur or camera shake during image capture;

(2) Sharpening: Enhances edge features to make defect boundaries more distinct;

(3) Brightness/Contrast Perturbation (RandomBrightnessContrast): Simulates varying lighting conditions;

(4) Color Distortion (HueSaturationValue): Simulates color temperature shifts and saturation variations;

(5) Horizontal Flip: Expands the sample space;

(6) Rotate: Simulates slight tilts during actual photography;

(7) Composite Enhancement Combinations: Combines multiple enhancement methods into three distinct combinations.

To prevent target disappearance after augmentation, a threshold of in_visibility=0.1 was set during the augmentation process.

Following augmentation, the dataset was expanded to 12, 000 images, significantly enriching the training sample space while enhancing the model's ability to detect minor defects and improve generalization performance. Augmented examples are shown in Figure 1.

Additionally, this paper enabled the Mosaic data augmentation strategy [6] built into the YOLOv8 training framework. Mosaic crops and stitches four distinct images into a new composite image, enabling simultaneous mixing augmentation at both the image and object levels to further enhance data diversity. As shown in Figure 2.

(a) Example before augmentation.

(b) Example after augmentation.

Figure 1 Demonstration of augmentation effects

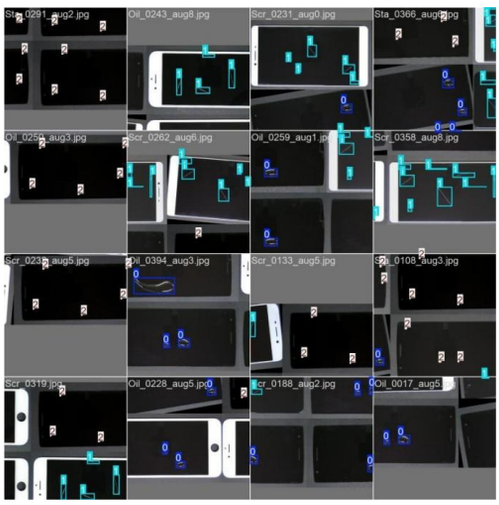

Figure 2 Mosaic-enhanced example

3 YOLOv8 Algorithm

3.1 YOLOv8 Overview

YOLOv8 is a one-stage detector that inherits the end-to-end detection mechanism from the YOLO series. Within this framework, the detector directly predicts object category probabilities and bounding box regression parameters on feature maps, eliminating the need for candidate box generation and cascaded classifiers. This end-to-end mechanism ensures computational efficiency [7].

In YOLOv8, the classification and regression branches are completely decoupled, a design inspired by FCOS and YOLOX. This decoupling allows both task branches to converge toward optimal solutions independently, theoretically reducing loss optimization instability and thereby improving convergence speed and detection accuracy.

YOLOv8's feature fusion component retains the dual-structure architecture of FPN (Feature Pyramid Network) [8] and PAN (Path Aggregation Network)[9]. Higher-level features possess stronger semantic abstraction capabilities, making them suitable for large object and category discrimination, while lower-level features contain more spatial and textural details, which are particularly critical for small object detection. YOLOv8 further integrates the C2f module [10], which enhances gradient flow through cross-stage partial connections and fast fusion paths. Compared to the C3 module, it shortens gradient propagation paths, reduces the risk of gradient vanishing, and minimizes redundant convolutional operations. This approach improves inference speed while maintaining accuracy.

Regarding bounding box regression and corresponding loss functions, YOLOv8 employs an IoU-based loss series combined with DFL (Distribution Focal Loss). The IoU-based loss series addresses the limitation of traditional L1/L2 regression in failing to capture overlap, directly optimizing the geometric overlap between predicted and ground-truth boxes to ensure geometric accuracy. DFL transforms the prediction of bounding box width and height into a distribution learning problem. This mechanism theoretically enhances the continuity and accuracy of bounding box predictions.

3.2 YOLOv8 Network Architecture

The YOLOv8 network comprises multiple layers. The backbone consists of convolutional layers, C2f modules, downsampling layers, and a Spatial Pyramid Pooling Function (SPPF) module, which progressively abstracts features from the input image. The detection head is responsible for fusing multi-scale features and outputting final detection results. It employs the FPN (Feature Pyramid Pooling) structure, combining upsampling, concatenation (Concat), and convolutions to achieve bottom-up feature enhancement and top-down feature propagation. Figure 3 shows the block diagram of YOLOv8 [11].

Furthermore, since YOLOv8 adopts an Anchor-Free approach, the anchors parameter has been removed from the YAML file. Multiple model parameter versions are now written into a single YAML file, allowing users to conveniently call YOLOv8 with different depth factors and width factors.

To achieve efficient end-to-end object detection, the YOLOv8 model employs a composite loss function that comprehensively evaluates three performance aspects: object classification accuracy, bounding box regression quality, and regression continuity. The overall loss function is expressed as follows:

$$L = \lambda_{box} L_{box} + \lambda_{cls} L_{cls} + \lambda_{dfl} L_{dfl} \tag{1}$$

Where `$\mathcal{L}_{cls}$` represents the object category classification loss, `$\mathcal{L}_{box}$` denotes the bounding box location regression loss, `$\mathcal{L}_{dfl}$` is the distributed focus loss used to enhance regression accuracy, and `$\lambda$` represents the weight coefficient.

3.2.1 Classification Loss $\mathcal{L}_{cls}$

YOLOv8 employs the BCEWithLogitsLoss (binary cross-entropy loss) for classification tasks, suitable for multi-label detection scenarios. The loss function is expressed as:

$$\mathcal{L}_{cls} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\sigma(\hat{y}_i)) + (1-y_i)\log(1-\sigma(\hat{y}_i)) \right] \tag{2}$$

where `N ` denotes the number of target classes, `$\hat{y}_i$ ` represents the predicted output for class `i `, `$y_i$ ` is the corresponding ground truth label, and `σ() ` denotes the Sigmoid function.

This loss function measures the error between the predicted classification probability and the true category.

3.2.2 Bounding Box Loss $\mathcal{L}_{box}$

The prediction accuracy of bounding box locations is measured using the Intersection over Union (IoU) series of loss functions. YOLOv8 defaults to the more stable and faster-converging Complete IoU (CIoU) [12] loss. Its form is as follows:

$$L_{CIoU} = 1 - IoU + \frac{\rho^2(b,b^g)}{c^2} + \alpha v \tag{3}$$

Where `b ` is the predicted bounding box, `$b^g$ ` is the ground truth bounding box, `$\rho^2(b,b^g)$ ` denotes the Euclidean distance between the centers of the predicted and ground truth boxes, `c ` is the diagonal length of the minimum bounding rectangle, `v ` is the aspect ratio consistency term, and ` α ` is the balancing factor.

CIoU comprehensively constrains the bounding box regression process, thereby improving detection accuracy and accelerating model convergence.

3.2.3 Distribution Focal Loss $\mathcal{L}_{dfl}$

To further enhance the continuity and accuracy of bounding box coordinate prediction, YOLOv8 introduces Distribution Focal Loss (DFL). This method transforms continuous coordinate prediction into learning probability distributions across a fixed number of bins.

For each coordinate (e.g.,x,y,w,h), the model outputs not a real value but R probabilities representing the likelihood of belonging to each bin. The final coordinate value is calculated as the weighted average across all bins:

$$\hat{x} = \sum_{i=0}^{R-1} i \cdot p_i \tag{4}$$

The loss function is the cross-entropy between the predicted probability distribution and the true distribution:

$$\mathcal{L}_{dfl} = -\sum_{i=0}^{R-1} q_i \log(p_i) \tag{5}$$

where `$q_i$ ` is the soft label or one-hot encoding corresponding to the true value, and `$p_i$ ` is the distribution probability predicted by the model.

Regarding activation functions, the YOLOv8 model employs SiLU (Sigmoid Linear Unit) as the primary activation function in both its backbone network and detection heads. This function strikes a favorable balance between performance and efficiency, serving as a smoother alternative to ReLU [13]. Its mathematical expression is:

$$\mathrm{SiLU}(x) = x \cdot \sigma(x) = \frac{x}{1+e^{-x}} \tag{6}$$

Where σ(x) represents the standard sigmoid function.

Compared to traditional activation functions, SiLU offers superior accuracy and higher gradient stability.

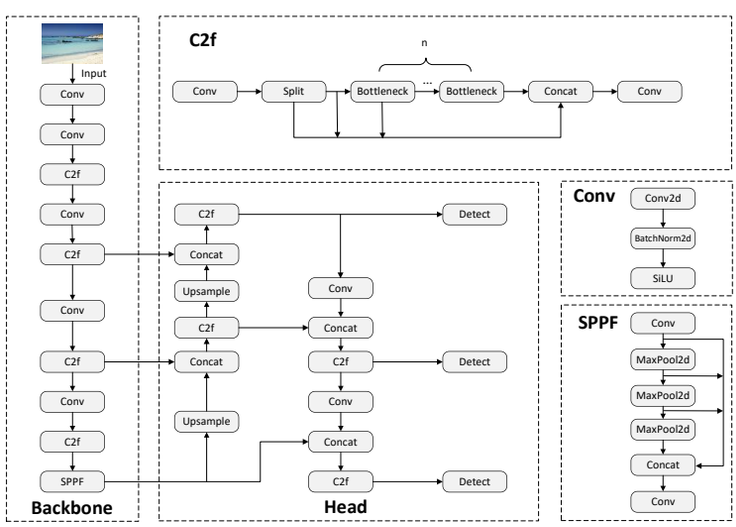

Figure 3 YOLOv8 Structural Diagram

4 Improved YOLOv8 Algorithm

4.1 Lightweight Backbone Network

In object detection tasks, the choice of backbone network directly determines a model's feature extraction capability and inference efficiency. As a mainstream single-stage detection framework, YOLOv8's default backbone structure exhibits strong feature fusion capabilities. However, it still carries significant computational overhead in terms of parameters and computational complexity, making it challenging to meet the stringent real-time and low-power requirements of mobile and embedded devices. MobileNetV4 [14] excels in cross-platform efficient inference, gradient flow optimization, and extended receptive field. Its MobileNetV4ConvSmall branch achieves significant parameter and floating-point operation reduction while maintaining robust feature representation, making it highly suitable as a backbone for detection networks in resource-constrained scenarios. Based on this consideration, this paper replaces YOLOv8's original backbone with MobileNetV4ConvSmall.

MobileNetV4 is a next-generation lightweight convolutional neural network introduced by Google in 2024. Compared to its predecessors MobileNetV1–V3, MobileNetV4 achieves significant improvements in module design, global modeling, and hardware adaptability. Key innovations include the introduction of the Universal Inverted Bottleneck (UIB) module and Mobile Multi-Query Attention (Mobile MQA).

UIB serves as the core module of MobileNetV4. It essentially generalizes and unifies the inverted bottleneck architecture from MobileNetV2. By flexibly adjusting the position of depthwise convolutions (DW) within the module, it can derive sub-architectures such as standard IB, ConvNeXt-like, FFN-like, and ExtraDW. Depthwise Convolution is an efficient computation method that significantly reduces computational load while preserving the receptive field. For high-level semantic feature extraction, MobileNetV4 introduces the Mobile MQA module to enhance global modeling capabilities. Mobile MQA effectively reduces complexity by decreasing the number of Key/Value pairs and incorporating downsampling operations.

MobileNetV4 offers multiple variants tailored to different tasks and hardware requirements. The Large branch targets high-precision tasks, the Medium branch balances accuracy and speed, the Small branch is optimized for low-power devices, the Hybrid branch integrates Mobile MQA in some higher layers for global modeling, while the Pure Conv branch uses purely convolutional structures to maximize hardware efficiency.

Among these, MobileNetV4ConvSmall represents the most lightweight variant. Its design significantly reduces model complexity by decreasing network depth and channel count, lowering the expansion ratio, and omitting attention modules. On embedded platforms, MobileNetV4ConvSmall maintains solid detection accuracy within limited computational resources. When serving as the backbone for YOLOv8, its advantages become particularly evident: on one hand, inference speed is significantly boosted; on the other, its compact size facilitates edge deployment.

4.2 Small Object Detection Optimization

In smartphone screen defect detection tasks, small targets typically exhibit characteristics such as low pixel coverage and poor texture contrast. To address these challenges, this paper introduces the SE (Squeeze-and-Excitation) attention module [15] into the backbone and feature fusion path of YOLOv8, enhancing the representation of small objects. This module improves channel discriminative power without altering the spatial resolution of feature maps, achieving stable gains with low parameter and computational costs.

The core concept of the SE module consists of two parts: Squeeze and Excitation. The former compresses each channel spatially via global average pooling, forming a compact global description that avoids statistical bias from relying solely on local receptive fields. The latter employs a two-layer fully connected network with a bottleneck structure to nonlinearly model channel importance. It generates normalized per-channel weights through a Sigmoid function, then backpropagates these weights to the original features for recalibration. Figure 4 illustrates the SE module architecture.

In implementation, adaptive global average pooling is employed to obtain channel statistics, while a bottleneck structure composed of two linear layers is used for weight estimation. The reduction ratior defaults to 16. When the number of channels C is small, appropriately reducing r prevents information loss due to excessive compression. Linear layer weights are initialized using He/Kaiming-type initialization to match ReLU nonlinearity. BatchNorm parameters are set to standard values during the initial phase to ensure numerical stability.

From a complexity perspective, the additional parameters in SE primarily stem from two fully connected matrices. Compared to the computational cost of the main convolutional layers—especially large-kernel convolutions—this overhead is typically secondary in magnitude, maintaining high cost-effectiveness under real-time constraints.

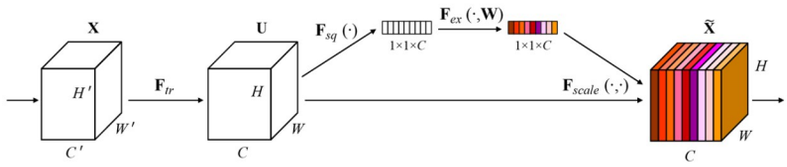

Figure 4 SE Module Architecture

4.3 Improved Network Model

To enhance YOLOv8's detection accuracy and deployment efficiency for mobile screen surface defect identification, this paper introduces two key improvements to the original YOLOv8s architecture. The core approach maintains YOLOv8's end-to-end single-stage detection framework while reducing model complexity through lightweight backbone replacement and attention mechanism integration, thereby increasing sensitivity to small objects.

First, in the backbone network component, this paper replaces YOLOv8s' backbone with MobileNetV4ConvSmall. This architecture achieves efficient convolutions and low-redundancy feature representation.

Second, for feature enhancement, this paper introduces a SE attention module between the backbone output and feature fusion layer. This module models inter-channel relationships and applies adaptive weighting to features. This approach enables the network to highlight defect-related salient features at an early stage of feature propagation.

In summary, YOLOv8s-M4SE preserves YOLOv8s' efficient single-stage detection capability while enhancing feature extraction and augmentation through lightweight backbone integration and attention mechanisms. This provides a more targeted network architecture for subsequent defect detection. The improved structure is shown in Table 1.

Table 1 YOLOv8s-M4SE Architecture

Layer Number	Input Source (from)	Module	Output Channels (params)	Parameters / Arguments
0	-1	MobileNetV4ConvSmall	-	Backbone Feature Extraction
1	-1	block.SPPF	128 → 512	[128, 512, 5]
2	-1	SEAttention	512	[512, 16]
3	-1	Upsample	512	[None, 2, 'nearest']
4	[1, 3]	Concat	-	Channel concatenation
5	-1	block.C2f	608 → 256	[608, 256, 1]
6	-1	Upsample	256	[None, 2, 'nearest']
7	[1, 2]	Concat	-	Channel concatenation
8	-1	block.C2f	320 → 128	[320, 128, 1] → P3 Features
9	-1	Conv (stride=2)	128	[128, 128, 3, 2] Downsampling
10	[1, 5]	Concat	-	Channel concatenation
11	-1	block.C2f	384 → 256	[384, 256, 1] → P4 Features
12	-1	Conv (stride=2)	256	[256, 256, 3, 2] Downsampling
13	[1, 2]	Concat	-	Channel concatenation
14	-1	block.C2f	768 → 512	[768, 512, 1] → P5 Feature
15	[12, 15, 18]	head.Detect	128/256/512	[3, [128, 256, 512]] Tri-scale prediction

5 Analysis of Experimental Results

5.1 Experimental Environment and Parameter Settings

The main parameter settings for this experiment are shown in Table 2.

Table 2 Training Parameters

Parameter Name	Parameter Value	Description
Learning Rate	0.001	Initial Learning Rate
Lower Limit of Learning Rate	0.01	Final learning rate
Iteration Count	200	Total Training Iterations
Warm-up rounds	5	First 5 rounds serve as warm-up to stabilize the training process
Input Image Size	640*640	Maintain consistent resolution
Batch size	16	Number of samples per iteration
Mosaic enhancement	1.0	Disable after 160 epochs

5.2 Evaluation Metrics

To comprehensively evaluate the performance of the proposed YOLOv8s-M4SE for mobile screen defect detection,

experiments employ metrics including Mean Average Precision (mAP), Precision, Recall, and computational complexity (FLOPs). These metrics characterize model performance across three dimensions: detection accuracy, coverage capability, and computational overhead.

In object detection evaluation, predicted bounding boxes must be matched one-to-one with ground truth boxes of the same category. Intersection over Union (IoU) measures the overlap between predicted and ground truth boxes, defined as follows:

$$IoU = \frac{area(B_{pred} \cap B_{gt})}{area(B_{pred} \cup B_{gt})} \tag{7}$$

A prediction is considered a positive match when IoU ≥ τ (τ is the threshold) and the categories match. Each ground truth box can match at most one prediction box; unmatched predictions are considered false positives (FP), and unmatched ground truth boxes are considered false negatives (FN). Correct negative class predictions are recorded as true negatives (TN).

The metrics used in this paper are defined as follows.

5.2.1 Precision

Precision measures the reliability of the model in classifying samples as positive. Higher precision indicates fewer false positives generated by the model, meaning fewer false alarms. The formula is:

$$Precision = \frac{TP}{TP + FP} \tag{8}$$

5.2.2 Recall

Recall measures a model's ability to detect true positive samples. A higher recall indicates fewer missed detections, reflecting the model's ability to identify genuine positives. The formula is:

$$Recall = \frac{TP}{TP + FN} \tag{9}$$

5.2.3 Average Precision (AP and mAP)

In object detection, Average Precision (AP) serves as an evaluation metric for a single category. It is calculated as follows: Given an IoU threshold τ , all detection results are first sorted by prediction confidence. The precision P(r) is then computed point-by-point for different recall thresholds r . A Precision-Recall (P-R) curve is plotted, and the area under this curve represents the AP value for that category:

$$AP = \int_0^1 P(r)\, dr \tag{10}$$

AP comprehensively reflects a model's ability to maintain high precision at different recall levels.

When the detection task involves multiple categories, the AP values for all categories are typically averaged to obtain the mean Average Precision (mAP):

$$mAP = \frac{1}{N} \sum_{c=1}^{N} AP_c \tag{11}$$

where `N ` denotes the number of classes, and `$AP_c$ ` represents the average precision of the `c `th class.

Depending on the IoU threshold, mAP is further categorized as:

$$mAP@0.5 = \frac{1}{N} \sum_{c=1}^{N} AP_c(\tau = 0.50) \tag{12}$$

This metric evaluates detection accuracy under more lenient overlap conditions.

$$mAP@0.5{:}0.95 = \frac{1}{N} \sum_{c=1}^{N} \frac{1}{10} \sum_{\tau \in \{0.50:0.05:0.95\}} AP_c(\tau) \tag{13}$$

This metric is more stringent, comprehensively reflecting the model's robustness and stability across varying localization accuracy requirements.

5.2.4 Computational Complexity (FLOPs, Floating Point Operations)

FLOPs represent the number of floating-point operations performed during a single forward inference pass. This metric measures computational complexity, directly impacting actual inference speed and hardware resource consumption. FLOPs serve as a critical indicator for assessing model deployability.

5.3 Analysis of Comparative Experimental Results

To validate the effectiveness of the proposed improvement strategy, this paper uses the original YOLOv8s as the baseline model. We sequentially introduce data augmentation, the lightweight backbone network MobileNetV4ConvSmall, and the SE channel attention module. Four experimental configurations are set: (A) Original YOLOv8s, (B) YOLOv8s + Data Augmentation, (C) YOLOv8s + MobileNetV4 + Data Augmentation, (D) YOLOv8s +

MobileNetV4 + SE + Data Augmentation (M4SE). Key metrics for each model group are compared in Table 3. A systematic analysis is provided below, focusing on detection performance.

(A) Original YOLOv8s (baseline model)

The baseline model achieves Precision of 94.9% and mAP@0.5 of 96.1%. The model demonstrates stable performance, accurately distinguishing targets from background. However, Recall is only 86.9% and mAP@0.5:0.95 is 81.1%, indicating significant missed detections, particularly for small-area, low-contrast defects where localization accuracy is insufficient.

(B) YOLOv8s + Data Augmentation

After applying diverse data augmentation, the model's Precision improved to 99.7%, Recall to 95.1%, mAP@0.5 to 98.3%, and mAP@0.5: 0.95 to 88.1%. The results demonstrate that the augmentation strategy effectively expands the sample distribution, enhancing the model's robustness and generalization capabilities, particularly improving recall performance for weakly textured "stain" objects.

(C) YOLOv8s + MobileNetV4 + Data Augmentation

After introducing MobileNetV4ConvSmall, computational complexity decreased from 28.7 GFLOPs to 19.0 GFLOPs, and the number of parameters dropped from 11.1M to 8.6M, achieving an overall efficiency improvement of approximately 30%. Precision and mAP@0.5 remained largely unchanged (99.4%, 98.3%), but Recall dropped to 94.7% and mAP@0.5: 0.95 to 86.4%. This indicates that MobileNetV4 suffers some loss in capturing fine-grained textures, leading to reduced localization accuracy in high IoU intervals.

(D) YOLOv8s + MobileNetV4 + SE + Data Augmentation (YOLOv8s-M4SE)

After adding the SE module to the lightweight architecture, Precision remained at 99.4%, Recall improved to 95.7%, mAP@0.5 and mAP@0.5:0.95 increased to 98.5% and 86.9%, respectively. These results demonstrate that the SE module enhances localization consistency in high IoU intervals. The most significant improvement was observed in detecting "stain" objects, as shown in Table 4. This demonstrates that the SE module notably enhances complex texture features. Adding only approximately 0.1M parameters and maintaining the same GFLOPs, this module achieves accuracy gains with negligible impact on inference speed. A limitation remains in the fine-grained feature loss of MobileNetV4, which warrants further optimization.

Table 3 Experimental Results Comparison

Model Configuration	Precision/%	Recall/%	mAP@0.5/%	mAP@0.5:0.95/%	Computational Load/GFLOPs	Parameters/10^6
YOLOv8s	94.9	86.9	96.1	81.1	28.7	11.1
Enhanced YOLOv8s	99.7	95.1	98.3	88.1	28.7	11.1
Enhanced + MobileNetV4	99.4	94.7	98.3	86.4	19.0	8.6
Enhanced + MobileNetV4 + SE	99.4	95.7	98.5	86.9	19.0	8.7

Table 4 Comparison of Experimental Results for the "Stain" Category

Defect Category	Model Configuration	Precision	Recall	mAP@0.5	mAP@0.5:0.95
stain	Original YOLOv8s	0.9975	0.7930	0.9743	0.6917
stain	Enhanced YOLOv8s	0.9993	0.9113	0.9797	0.7483
stain	Enhanced + MBV4	0.9920	0.9073	0.9813	0.7388
stain	Enhanced + MBv4 + SE	0.9971	0.9215	0.9842	0.7489

Comprehensive comparison reveals that the YOLOv8s-M4SE model achieves a good balance across accuracy, recall, and inference efficiency. These results demonstrate efficient and accurate detection of surface defects on smartphone screens.

6 Conclusion

This paper addresses the challenges of high small object miss rates and excessive model complexity in smartphone screen surface defect detection. Based on the YOLOv8 framework, this paper proposes an improved model, YOLOv8s-M4SE, which integrates a lightweight backbone with a channel attention mechanism. By refining data augmentation strategies, backbone architecture, and feature channel modeling, the model achieves a favorable balance between detection accuracy and computational efficiency, validating its effectiveness and feasibility for smartphone screen defect detection.

During data preprocessing, a multidimensional data augmentation strategy was designed, incorporating operations such as blurring, sharpening, brightness perturbation, color bias, rotation, and mosaic stitching. This significantly enhances the model's adaptability, with experiments demonstrating its effective improvement in generalization capability.

Regarding model architecture optimization, the original YOLOv8s backbone was replaced with MobileNetV4 ConvSmall, achieving a lightweight network design. This substitution reduced computational load from 28.7 GFLOPs to

19.0 GFLOPs and decreased parameters from 11.1M to 8.6M, substantially improving inference efficiency with only a slight performance degradation. Second, the embedded SE channel attention module enhances responses to defect-related features, effectively improving the precision and recall of weakly textured small targets like "stains".

Experimental results demonstrate that the improved YOLOv8s-M4SE model achieves outstanding performance on public smartphone screen defect datasets, striking an effective balance between lightweight implementation and high-precision detection.

References

[1] R. Khanam, M. Hussain, R. Hill, and P. Allen, "A Comprehensive Review of Convolutional Neural Networks for Defect Detection in Industrial Applications," in IEEE Access, vol. 12, pp. 94250–94295, 2024.

[2] G. Yang, W. Xiong, J. Lei, and K. Yang, "Blade-YOLOv8: Improved YOLOv8 for Wind Turbine Blade Defect Detection," 2024 4th Power System and Green Energy Conference (PSGEC), Shanghai, China, 2024, pp. 209-213.

[3] L.-G. Wu, Z. Zhang, L. Chen, and Q. Zhou, "SSB YOLOv8's Method for Detecting Lump Coal in Mine Conveyors," 2024 43rd Chinese Control Conference (CCC), Kunming, China, 2024, pp. 8798-8803.

[4] H. Liu, R. Hu, H. Dong, and Z. Liu, "SFC-YOLOv8: Enhanced Strip Steel Surface Defect Detection Using Spatial-Frequency Domain-Optimized YOLOv8," in IEEE Transactions on Instrumentation and Measurement, vol. 74, pp. 1-11, 2025, Art no. 9700111.

[5] C. Kamardi et al., "Classification of Alzheimer's Disease using Random Oversampling and Albumentations on Convolutional Neural Network," 2023 Eighth International Conference on Informatics and Computing (ICIC), Manado, Indonesia, 2023, pp. 1-6.

[6] C. Wang, Z. Zhou, and Z. Chen, "An Enhanced YOLOv4 Model With Self-Dependent Attentive Fusion and Component Randomized Mosaic Augmentation for Metal Surface Defect Detection," in IEEE Access, vol. 10, pp. 97758-97766, 2022.

[7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 779-788.

[8] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature Pyramid Networks for Object Detection," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 936-944.

[9] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path Aggregation Network for Instance Segmentation," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 2018, pp. 8759-8768.

[10] H. Li, A. Wu, Z. Jiang, F. Liu, and M. Luo, "Improving Object Detection in YOLOv8n with the C2f-f Module and Multi-Scale Fusion Reconstruction," 2024 IEEE 6th Advanced Information Management, Communications, Electronic and Automation Control Conference (IMCEC), Chongqing, China, 2024, pp. 374-379.

[11] R. Varghese and S. M., "YOLOv8: A Novel Object Detection Algorithm with Enhanced Performance and Robustness," 2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS), Chennai, India, 2024, pp. 1-6.

[12] S. Du, B. Zhang, P. Zhang, and P. Xiang, "An Improved Bounding Box Regression Loss Function Based on CIOU Loss for Multi-scale Object Detection," 2021 IEEE 2nd International Conference on Pattern Recognition and Machine Learning (PRML), Chengdu, China, 2021, pp. 92-98.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 2015, pp. 1026-1034.

[14] Qin, D., Leichner, C., Delakis, M., Fornoni, M., Luo, S., Yang, F., Wang, W., Banbury, C.R., Ye, C., Akin, B., Aggarwal, V., Zhu, T., Moro, D., & Howard, A. (2024). MobileNetV4 - Universal Models for the Mobile Ecosystem. European Conference on Computer Vision.

[15] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-Excitation Networks," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 42, no. 8, pp. 2011-2023, 1 Aug. 2020.