Journal of Electronic Testing (2024) 40:645–656
https://doi.org/10.1007/s10836-024-06146-2

RESEARCH

YOLOv8-TDD: An Optimized YOLOv8 Algorithm for Targeted Defect Detection in Printed Circuit Boards

Gao Yunpeng[1] · Zhang Rui[1] · Yang Mingxu[2] · Fahad Sabah[3]

Received: 3 June 2024 / Accepted: 7 October 2024 / Published online: 5 November 2024

Abstract
An enhanced approach for detecting defects in Printed Circuit Boards (PCBs) using a significantly improved version of the YOLOv8 algorithm is proposed in this research, the proposed method is referred to as YOLOv8-TDD (You Only Look Once Version8-Targeted Defect Detection). This novel approach integrates cutting-edge components such as Swin Transformers, Dynamic Snake Convolution (DySnakeConv), and Biformer within the YOLOv8 architecture, aiming to address and overcome the limitations associated with traditional PCB inspection methods. The YOLOv8-TDD adaptation incorporates Swin Transformers to leverage hierarchical feature processing with shifted windows, enhancing the model's efficiency and capability in capturing complex image details. Dynamic Snake Convolution is implemented to dynamically adapt filter responses based on the input feature maps, offering tailored feature extraction that is highly responsive to the varied textures and defects in PCBs. The Biformer, with bidirectional processing capability, enriches the model's contextual understanding, providing a comprehensive analysis of the PCB images to pinpoint defects more accurately. Experimental results demonstrate that YOLOv8-TDD model, achieves a precision of 97.9%, a mean Average Precision (mAP0.5) of 95.71%. This enhanced model offers significant potential for practical applications in PCB manufacturing, promising to elevate quality control standards through more reliable defect detection.

Keywords Defect Detection · Computer vision · Printed Circuit Boards · YOLO · Deep Learning · Attention Mechanism

1 Introduction

Printed Circuit Boards (PCBs) are essential for electronic devices, serving as the backbone that interconnects various electronic components. Quality assurance in PCBs is crucial for the reliability and functionality of these devices. However, manufacturing defects like scratches, cracks, and soldering issues can degrade performance or cause complete failure. Traditional PCB inspection methods, such as manual visual inspection and automated optical inspection (AOI), are often time-consuming, labor-intensive, and susceptible to human error.

As PCBs become more integrated and miniaturized, detecting smaller and more complex defects becomes increasingly challenging. Implementing comprehensive defect detection strategies is critical for improving product quality and reducing manufacturing costs [1]. Traditional defect detection methods include manual visual inspection, electrical testing, and AOI [2]. Manual inspection is increasingly inadequate due to the precision required in modern PCB production, characterized by poor detection reliability and efficiency. Electrical testing, while effective, requires elaborate setups and is limited in evaluating multi-layered PCBs, with the risk of secondary damage.

AOI uses machine vision technology and sophisticated image processing algorithms to provide stable, accurate, and fast defect detection without compromising PCB integrity [3, 4]. Advances in deep learning have led to the development of contactless automatic detection techniques.

Responsible Editor: V. D. Agrawal.

✉ Zhang Rui
zhangrui@bigc.edu.cn

1 Department of Mechanical and Electrical Engineering, Beijing Institute of Graphic Communication, Beijing 102600, China

2 Department of Mechanical and Electrical Engineering, Beijing Information Science and Technology University, Beijing 100192, China

3 Department of Information Technology, Beijing University of Technology, Beijing 100124, China

Detection networks are categorized into one-stage and two-stage methods. One-stage networks, such as Single Shot Detector (SSD) and YOLO [5, 6], predict defect locations and categories directly after feature extraction. Two-stage networks, like R-CNN and its variants [7–9], first generate defect proposals and then refine these into precise detections. While two-stage networks are thorough, they are slower due to extensive candidate frame generation. One-stage networks integrate training and detection, enhancing speed and efficiency.

This paper refines the YOLOv8 model for real-time industrial applications [10]. The Swin Transformer V2 addresses three major challenges in large-scale visual model training: instability, resolution discrepancies, and the scarcity of annotated datasets. It introduces hybrid cosine attention mechanisms, logarithmic space techniques for resolution scaling, and SimMIM, a self-supervised pretraining method, reducing reliance on extensive labeled data. Swin Transformer V2 also integrates SwinV2 CSPB modules to enhance efficiency [11]. In machine learning, the attention mechanism is crucial for improving model performance. Shuffle Attention (SA) advances this by permuting and resequencing input sequences before computing attention weights, enhancing computational efficiency and model generalization [12].

This research paper is structured as follows: Sect. 2 reviews existing literature. Section 3 details the techniques incorporated into the enhanced defect detection method. Section 4 presents the empirical evaluation of the proposed method. Sections 5 and 6 conclude with the key findings of the research.

2 Related Work

The traditional approach to detecting visual anomalies in artificial systems often involves high costs, low efficiency, and susceptibility to errors. To address this, leveraging the electrical properties of components has been proposed for defect detection in printed circuit boards (PCBs) through a semi-automatic, manual method that incorporates online and functional testing [13]. Researchers have pursued various enhancements to this approach, such as using wavelet transforms to compress images and reduce memory and computational demands [14]. Additionally, traditional machine learning algorithms have been applied to improve defect detection [15], alongside the development of low-complexity neural networks and machine vision techniques [16]. Other innovative strategies include Fourier image reconstruction for identifying minute defects [17] and ultrasonic laser thermal imaging for real-time defect detection [18]. Despite potential cost reductions, these methods remain limited due to factors like non-reusability of the testing process, high equipment costs, and complex programming requirements.

Machine vision detection techniques have emerged as an effective alternative, gaining traction in modern industrial applications [19]. Specific image segmentation techniques used include threshold segmentation, edge segmentation, and region segmentation. For instance, Ardhy et al. utilized adaptive Gaussian threshold segmentation for rapid defect detection with minimal parameter adjustments, though effectiveness varied across different areas, especially those affected by lighting [20]. Baygin et al. applied the Hough transform for edge detection combined with the Canny operator to improve detection efficiency [21]. Ma et al. enhanced the region growth algorithm for better defect identification outcomes [22]. While these techniques offer improved detection capabilities, they often require manual tuning of model parameters, leading to suboptimal accuracy and efficiency.

Automated optical inspection (AOI) methods have been highlighted for their superior accuracy. However, AOI systems are sensitive and governed by stringent parameter-setting requirements, often leading to missed defects and necessitating subsequent manual reviews [23]. Conversely, deep learning technologies have advanced rapidly, offering promising alternatives for defect detection.

DenseNet has been used effectively due to its architecture that densely connects all layers both forwards and backwards, facilitating feature reuse and reducing the need for many parameters and computational resources [24]. Huang et al. designed a convolutional neural network that enhances detection accuracy and efficiency by connecting each layer in a sequential, feedforward manner [25].

Deep learning algorithms are known for their strong non-linear capabilities, robustness, and suitability for complex scenarios, contributing to superior performance over conventional machine vision methods. For example, a method proposed in [26] achieved an accuracy rate of 96.91%. Geng et al. used focal loss with a ResNet50 backbone to increase detection accuracy to 96.65% [27]. Ding et al. developed TDD-net, a network tailored for detecting tiny PCB defects, employing a multi-scale fusion strategy and online hard example mining to improve the certainty of region-of-interest (ROI) proposals, resulting in a detection accuracy of 98.90% [28].

Hu et al. introduced UF-Net, preserving more defect target information through the Skip Connect method, achieving a detection accuracy of 98.6% [29]. Li et al. enhanced the mean Average Precision (mAP) to 98.71% by integrating a residual structure unit CSP into the YOLOv4 algorithm's trunk [30]. Wang et al. proposed a lightweight model using the ShuffleNetV2 structure within the YOLOv5 backbone, reaching an accuracy of 95% [31]. YOLOv7 has emerged as a benchmark in target detection algorithms, surpassing previous iterations of the YOLO series in terms of both

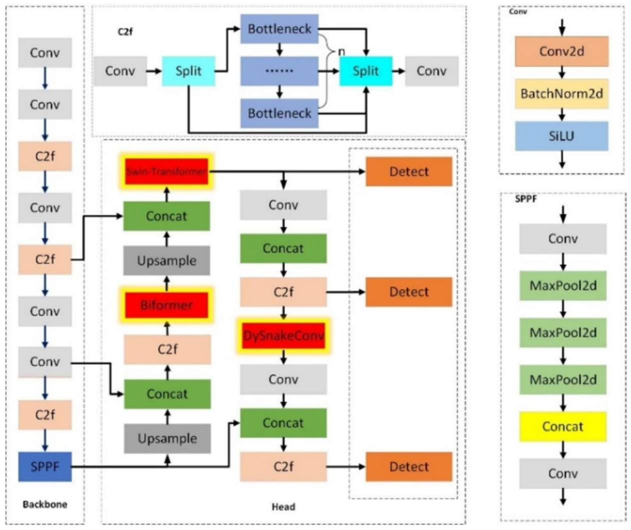


Fig. 1 Proposed Architecture

detection speed and accuracy, showcasing the ongoing evolution and effectiveness of deep learning in practical applications.

This manuscript introduces a novel methodology for detecting defects in PCBs, based on an extensive analysis of the aforementioned algorithms. The key enhancements of this methodology involve integrating a YOLOv8-TDD architecture into the core network framework. This integration aims to augment image resolution, reinforce model stability, and refine local feature extraction within PCB imagery, thereby facilitating more accurate anomaly identification.

3 Proposed Methodology

The architecture depicted in Fig. 1 incorporates a series of specialized components designed to process and transform input data for object detection tasks. Here's a detailed explanation of the workflow and functionality of each component within the proposed architecture:

3.1 Existing Architecture

Backbone The backbone is responsible for extracting features from the input image. It consists of a series of convolutional layers (Conv) that progressively capture higher- level features. The convolution operation in a layer can be mathematically described as:

$$O_{ij} = \sum_{m} \sum_{n} I_{(i+m)(j+n)} K_{mn} \tag{1}$$

Here, O is the output feature map, I is the input feature map, K is the convolutional kernel, and m, mn, n are the indices that run over the kernel size.

Spatial Pyramid Pooling-Fast (SPPF) Located at the end of the backbone, SPPF is a layer designed to aggregate context information by applying different-sized pooling operations. It helps the network maintain spatial hierarchies and improves the robustness of the feature maps against object scale variations. The SPPF can be represented by a pooling operation over varying window size. Let's denote $P_l(I)$ as the pooling operation at level l of the pyramid:

$$S_l = P_l(I) \tag{2}$$

where S_l is the pooled output at level l, and I is the input feature map to the SPPF layer.

C2f The blocks, marked as C2f, are special feature processing layers. It performs operations that enhance the channel-wise or spatial features before passing them to subsequent layers or before fusing them with other feature maps.

Conv Layers Standard convolutional layers are used throughout the architecture to apply learned filters to the incoming feature maps. These layers are the building blocks of the network, responsible for detecting patterns such as edges, textures, and more complex shapes in deeper layers.

Split and Bottleneck The 'Split' operation implies a divergence in the pathway, where the feature map is divided, and separate transformations are applied in parallel paths.

Upsample Upsampling is critical in object detection to allow for precise localization of objects after the resolution has been reduced by previous convolutional and pooling layers. For nearest-neighbor up-sampling, mathematically:

$$U(x, y) = I(\lfloor x/s \rfloor, \lfloor y/s \rfloor) \tag{3}$$

where U is the up-sampled image, I is the input image, x, y are the coordinates in the up-sampled image, and s is the scaling factor.

Concat The 'Concat' operation merges feature maps from different sources. This is a crucial step in object detection architectures, as it allows the network to consider information from various stages of processing, combining low-level details with high-level semantic information. Concatenation of feature maps along the channel dimension can be represented as:

$$C = concat\left(F_1, F_2, \ldots, F_n\right) \tag{4}$$

where C is the concatenated output, and $F_1, F_2, ..., F_n$ are the feature maps being concatenated.

Head and Detect The'Head' includes the final layers of the network that make predictions based on processed feature maps. The'Detect' blocks are specialized layers/modules responsible for producing the final object detection output, which includes determining the presence of objects, their locations, and their categories. The encoder (Backbone and SPPF) captures the context, while the decoder (formed by Upsample and Concat operations, as well as the attention mechanisms) reconstructs the spatial details required for accurate localization.

Attention Mechanism An attention mechanism can be represented as:

$$Attention(Q, K, V) = Softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{5}$$

where *Attention* is the attention output, Q, K, and V represent the queries, keys, and values in the attention mechanism, respectively, and d_k is the scaling factor. Transformers (general form): Transformer layers contain a multi-head self-attention mechanism and can be described as:

$$MHSA(X) = Concat(head_1, head_2, ..., head_h)W^O \tag{6}$$

$$head_i = Attention\left(XW_i^Q, XW_i^K, XW_i^V\right) \tag{7}$$

where X is the input, W_i^Q are weight matrices for the i *th* attention head, W^O is the output weight matrix, and h is the number of heads.

3.2 Newly Added Architecture

3.2.1 Swin Transformers [32]

As shown in Fig. 2, Swin Transformers are a type of vision transformer that computes self-attention within local windows, which are shifted across layers. This architecture enables hierarchical feature representation, similar to that of CNNs, while the local windowing mechanism reduces computational complexity. By shifting the windows in subsequent layers, Swin Transformers allow for cross-window connections and a broader receptive field, enhancing feature extraction over multiple layers. Compared to traditional global self-attention, Swin Transformers reduce computational costs while preserving the global feature extraction capabilities of Transformers.

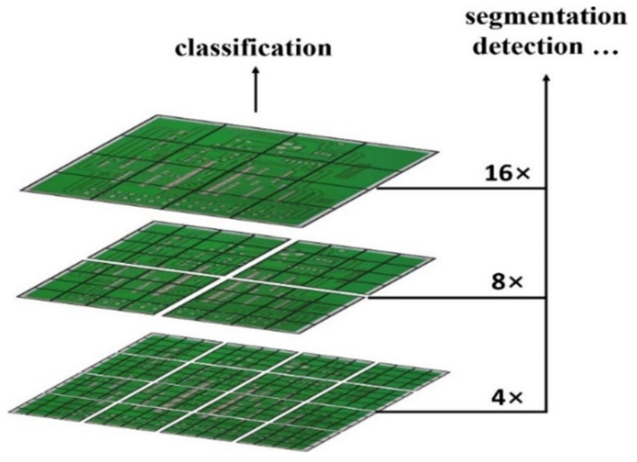


Fig. 2 Swin Transformer Architecture

Swin Transformers exhibit strong feature extraction, particularly with large-scale data, and their window-based mechanism enables better capture of fine details in complex scenes, potentially improving detection accuracy. Therefore, we integrate Swin Transformers into YOLOv8 to enhance its ability to extract features from large-scale PCBs data, capture finer details in complex scenarios, and ultimately improve detection performance.

$$Attention(Q, K, V) = Softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{8}$$

Here, Q, K, and V are the query, key, and value matrices computed within each window, and d_k is the dimensionality of the keys.

3.2.2 DySnakeConv [33]

As shown in Fig. 3, DySnakeConv is a dynamic convolutional architecture designed to capture object edges and contours. It dynamically adjusts the shape of convolutional kernels to better align with object boundaries, making it highly effective for detecting objects with complex or irregular shapes. In object detection, edge features are crucial for accurate localization. By enhancing YOLOv8's ability to detect and localize objects with varying shapes and boundaries, DySnakeConv improves performance, particularly for irregularly shaped objects.

Therefore, we incorporate DySnakeConv to better capture the edges and contours of defective parts on PCBs, thereby improving detection accuracy.

The general form could be expressed as:

$$y(x) = \sum_{i=1}^{N} a_i(x) \times f_i(x) \tag{9}$$

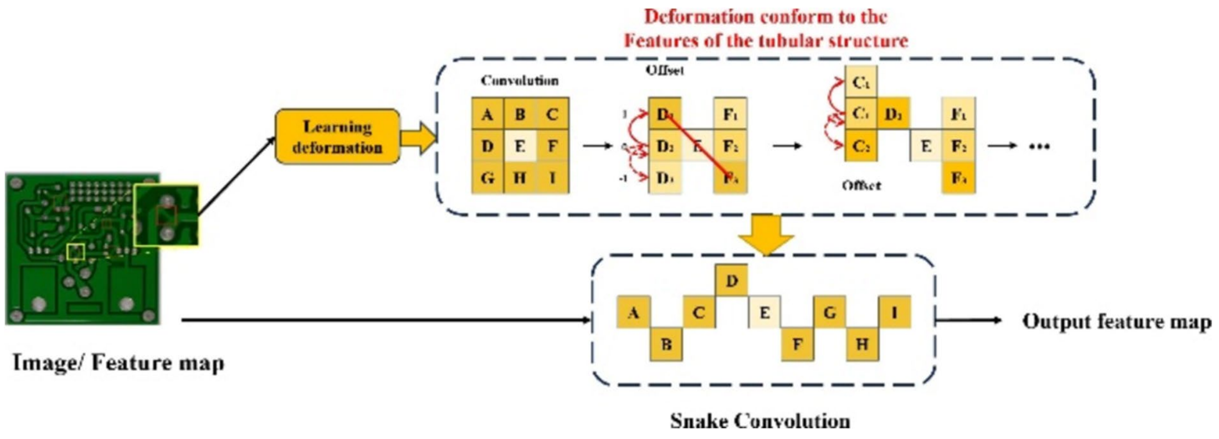


Fig 3 Dynamic Snake Convolution Architecture

where $y(x)$ is the output, $f_i(x)$ represents the convolutional operations with different kernels, and α_i is the input-dependent weights.

3.2.3 Biformer [34]

As illustrated in Fig. 4, Biformer is a bidirectional Transformer architecture that leverages the strengths of both self-attention mechanisms and convolutional neural networks to enable efficient feature fusion and information exchange. This design allows it to capture both long-range dependencies and local details effectively.

The bidirectional nature of Biformer enhances multi-scale object detection and feature fusion in YOLOv8. By facilitating the integration of global and local information, it improves the model's accuracy in detecting PCBs of varying sizes and scales. Bidirectional transformers generally compute representations as follows:

$$H_{bi} = \left[\overrightarrow{H};\overleftarrow{H}\right] \tag{10}$$

Wher H_{bi} is the combined bidirectional representation, $\overrightarrow{H}$ is the forward pass representation, and $\overleftarrow{H}$ is the backward pass representation.

3.2.4 CIoU, SIoU, EIoU (Different IoU loss functions)

1. CIoU (Complete IoU)

CIoU improves upon standard IoU by considering not just the overlap between bounding boxes but also their aspect ratio and center point distance. It helps optimize the alignment between predicted and ground truth boxes, especially in cases where objects have different shapes or orientations. Integrating CIoU into YOLOv8 can improve localization accuracy by ensuring that predictions are better aligned with ground truth, particularly for objects with varying aspect ratios or off-center placements. This helps reduce bounding box regression errors.

2. SIoU (Skew IoU)

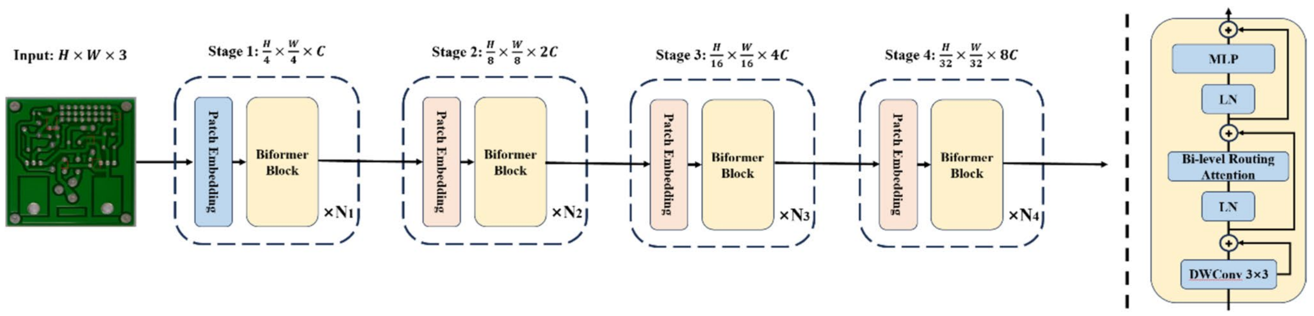

Input: $H \times W \times 3$
Stage 1: $\frac{H}{4} \times \frac{W}{4} \times C$
Stage 2: $\frac{H}{8} \times \frac{W}{8} \times 2C$
Stage 3: $\frac{H}{16} \times \frac{W}{16} \times 4C$
Stage 4: $\frac{H}{32} \times \frac{W}{32} \times 8C$
Patch Embedding
Biformer Block
×N1
×N2
×N3
×N4
MLP
LN
Bi-level Routing Attention
LN
DWConv 3×3

Fig. 4 Biformer Architecture

SIoU introduces a penalty for misalignment between bounding boxes by focusing on their geometric properties and orientation. It accounts for the skewness and angle of the predicted box relative to the ground truth, which is useful in detecting rotated or irregularly shaped objects. SIoU can improve the detection of objects that are not aligned with the horizontal or vertical axis, making YOLOv8 more robust in scenarios where object rotation and skewness are significant. This can lead to better performance in detecting rotated or slanted objects.

3. EIoU (Efficient IoU)

EIoU enhances IoU by focusing on the distance between bounding box centers, aspect ratio, and overlap area, similar to CIoU but with a more efficient approach to penalize mismatches. It reduces the computational burden while maintaining high accuracy in bounding box regression. EIoU can offer improved computational efficiency for bounding box regression while maintaining high localization accuracy. This makes it a good fit for real-time detection tasks where performance and speed are critical, enhancing YOLOv8's efficiency in processing large-scale datasets.

$$L_{CIoU} = 1 - IoU + \frac{\rho^2(b, b^{gt})}{c^2} + av \tag{11}$$

where ρ is the Euclidean distance between the box centers b and b^{gt}, c is the diagonal distance of the smallest enclosing box covering both the predicted and ground truth boxes, v is a measure of consistency of aspect ratio, and a is a trade-off parameter.

4 Experiments

This section discuss the dataset, performance measures and the environment for simulations.

4.1 Dataset Used

The dataset we used for our experiments is PCB (Printed Circuit Boards) images [35] have different kinds of defects. It has training and validation size 8702 out of which the training size is 7831.

As shown in Table 1, each category in the dataset represents a different class of fault that the machine learning model will learn to detect. The numbers, which indicate the instances of each fault type, show that a well-distributed set of examples is available. This distribution is advantageous as it allows the model to learn effectively from a sufficient number of examples. Table 1 demonstrates that the dataset is well-prepared and categorized for PCB fault detection, with text files generated to be compatible with machine learning frameworks such as TensorFlow or PyTorch. The relatively balanced class distribution is beneficial for training robust models that can generalize well across various PCB fault types.

Table 1 Dataset Classes and their Counts

Class	Count
False Copper	4348
Missing Hole	4321
Mouse Bite	7075
Open Circuit	7275
Pin Hole	16684
Scratch	4918
Short Circuit	6053
Spur	15682

As shown in Fig. 5, the following are the details regarding types of PCB faults:

- FalseCopper: (Fig. 5.a) FalseCopper refers to defects where copper is present where it should not be, such as unintended copper traces on the PCB.
- MissingHole: (Fig. 5.b) These are the defects characterized due to the absence of a hole that should have been drilled into the PCB.
- MouseBite: (Fig. 5.c) 'mouse bite' is a term used in PCB manufacturing to describe a series of small notches that may appear along the edge of a PCB.
- OpenCircuit: (Fig. 5d) OpenCircuit occurs when there is a break in the circuitry, preventing current from flowing as intended.
- PinHole: (Fig. 5e)PinHoles are small holes in the solder mask or on the PCB that are unplanned and can cause issues in the circuitry.
- Scratch: (Fig. 5f). Scratches on the PCB can lead to exposed copper and potential short-circuits, an example of this type of fault can be seen in Fig. 5f.
- ShortCircuit: (Fig. 5g), ShortCircuit happens when there is an unintended connection between two points in the circuit, causing a short.
- Spur: (Fig. 5h) A spur is an unwanted copper trace that comes off of a pad or a trace.

4.2 Performance Measures Used

P (Precision) The ability of the model to identify only the relevant objects. Higher is better. Precision is the ratio of correctly predicted positive observations to the total predicted positive observations.

$$P = \frac{TP}{TP + FP} \tag{12}$$

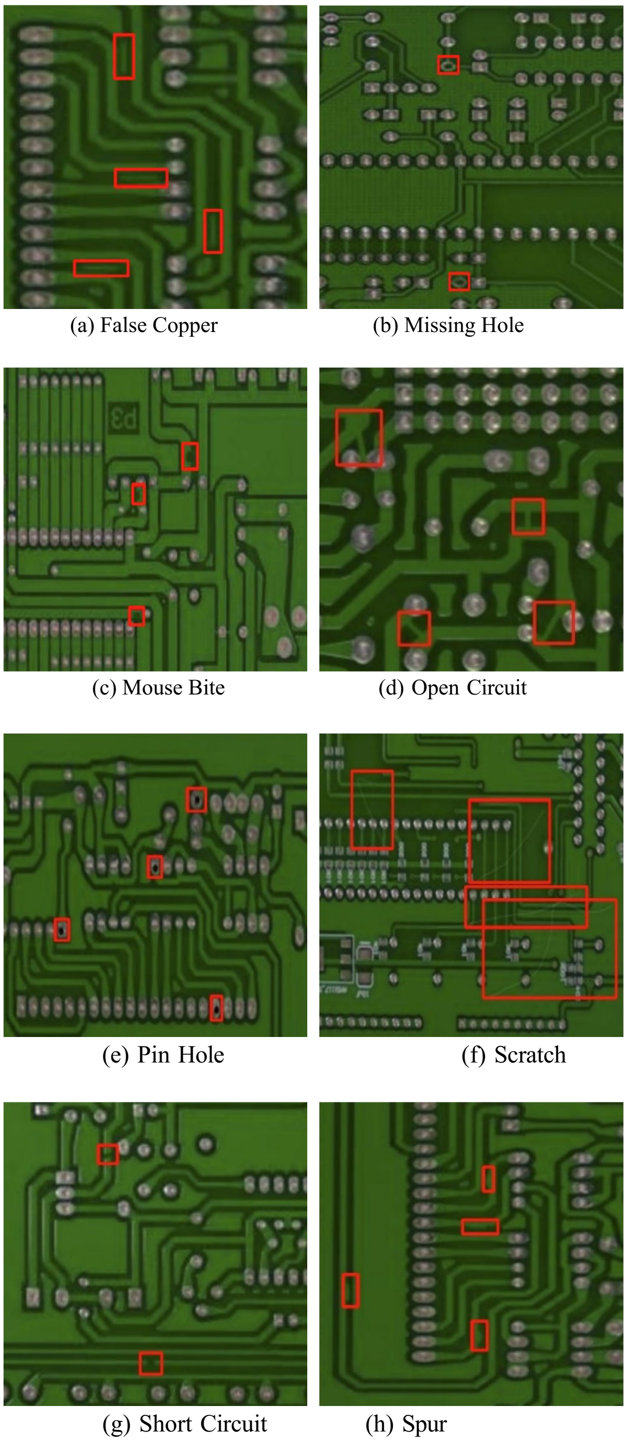


Fig. 5 PCB fault types sample images (**a**) False Copper (**b**) Missing Hole (**c**) Mouse Bite (**d**) Open Circuit (**e**) Pin Hole (**f**) Scratch (**g**) Short Circuit (**h**) Spur

where *TP* is true positives and *FP* is false positives.

R (Recall) The ability of the model to identify all relevant objects. Higher is better. Recall is the ratio of correctly predicted positive observations to all observations in actual class.

$$R = \frac{TP}{TP + FN} \tag{13}$$

where *TP* is true positives and *FN* is false negatives.

mAP0.5 (mean Average Precision at IOU = 0.5) This is the average precision at an Intersection Over Union (IOU) threshold of 0.5. Higher is better

$$mAP0.5 = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{14}$$

Where N is the number of classes and AP_i is the average precision for class i at IOU = 0.5.

mAP0.5:0.9 This is the mean Average Precision calculated over different IOU thresholds from 0.5 to 0.9. Higher is generally better, indicating good performance across a range of IOU thresholds

$$mAP0.5 : 0.9 = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{T_i} \sum_{t=0.5}^{0.9} AP_i^t \tag{15}$$

Where T is the number of thresholds, AP_i^t is the average precision for class i at threshold t.

FPS (Frames Per Second) The number of frames the model can process in one second. Higher is better, indicating a faster model.

$$FPS = \frac{(Number\ of\ processed\ frames)}{(Time\ in\ Seconds)} \tag{16}$$

4.3 Simulation Environment

Windows11, Python version 3.10.13, torch-2.0.0, CUDA-11.8, cudnn-8.4.1 (NVIDIA GeForce RTX 3060 Ti), epoch = 100, batch = 16, Imgsz = 640, Lr = 0.01, Momentum = 0.937. Weight decay = 0.0005.

5 Results

5.1 Attention Mechanism

The Table 2 lists the results of an experiment comparing the performance of different attention mechanisms in the same computational model(YOLOv8) for a PCBs computer vision task given the context of the mAP% metric.

CA, or Channel Attention, focuses on the inter-channel relationships of features. ECA, or Efficient Channel

Table 2 Attention Mechanisms Performance Comparison

Attention	P	R	mAP0.5	FPS
YOLOv8 + CBAM	0.883	0.843	0.882	317.39
YOLOv8 + DAT	0.883	0.874	0.912	244.93
YOLOv8 + CA	0.895	0.881	0.915	434.78
YOLOv8 + ECA	0.918	0.882	0.923	377.78
YOLOv8 + Biformer	0.939	0.896	0.938	333.33

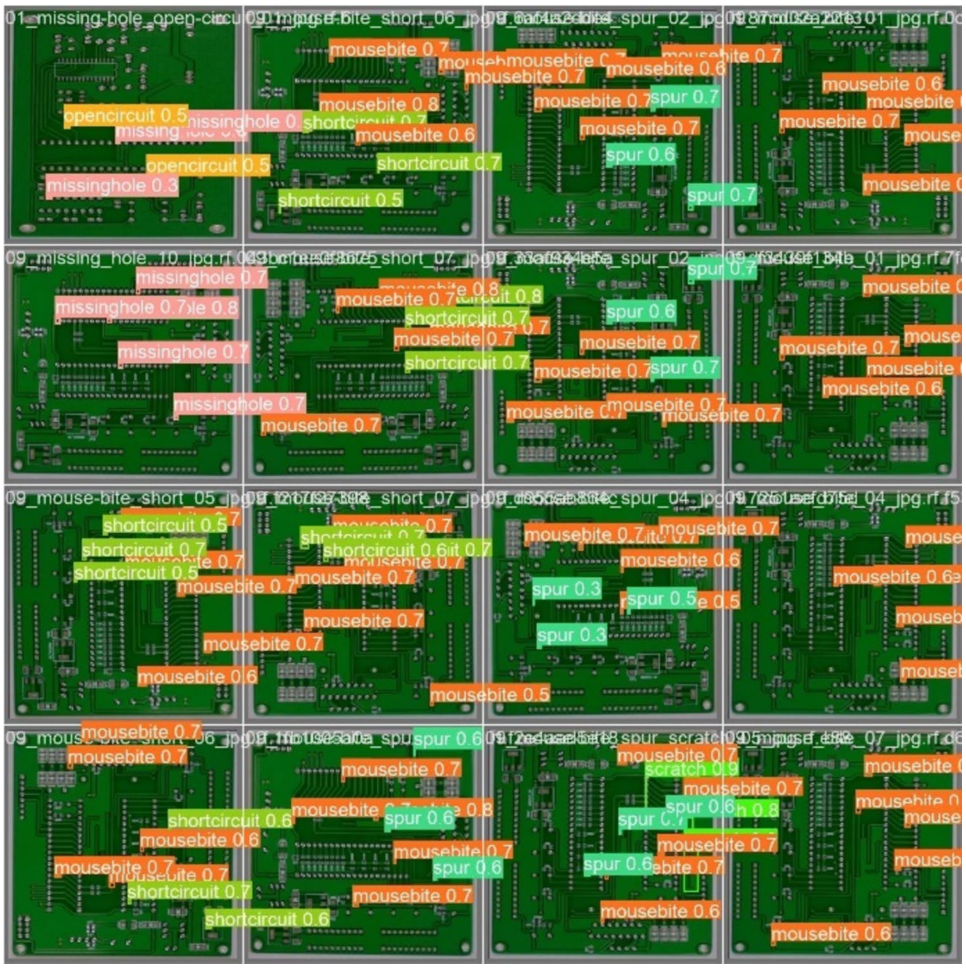
Attention, is a computationally more efficient version of Channel Attention. CBAM, or Convolutional Block Attention Module, integrates both spatial and channel-wise attention within a single module. DAT, or Dual Attention, combines two different attention mechanisms that address various aspects of the input data or stages of the processing pipeline, such as self-attention with cross-attention or local attention with global attention. Biformer refers to a bidirectional transformer-based attention mechanism.

The results indicate that Biformer stands out with the highest precision (0.939), recall (0.896), and mAP0.5 (0.938), making it the best performer in detection accuracy, though its FPS (333.33) is lower than CA (434.78) and ECA (377.78). CA and ECA also demonstrate strong performance, with ECA having slightly better precision (0.918) and mAP0.5 (0.923) than CA, but CA maintains the fastest processing speed. CBAM and DAT, while improving detection metrics over simpler models, lag behind in both speed and some accuracy measures compared to the other attention mechanisms. Overall, Biformer offers the best detection performance, while CA and ECA provide an excellent balance of accuracy and speed. So we choose Biformer as the attention mechanism to be added to YOLOv8-TDD.

5.2 Module Ablation Experiment

Figure 6 shows the validation batch results, here are some key observations from this Fig; The confidence scores seem to be consistently high across various defect types, which is a good indicator of the model's reliability. Some defects appear to have multiple overlapping bounding boxes, which might indicate a level of uncertainty in the model when localizing the defect. The readability of labels is crucial for practical applications. Figure 6 shows clear labeling of

Fig. 6 Validation Batch Results



detected defects, which is beneficial for end-users such as quality control engineers to verify and take action on the detections.

The Table 3 compares different configurations of a YOLOv8-based object detection system, with variations including additional architectural features or techniques like Swin Transformer, DySnakeConv, Biformer, and various inter- section over union (IoU) loss functions such as CIoU, SIoU, and EIoU.

Here's a detailed look at what each column signifies and an analysis of the results; it can be seen in the Table 4, The addition of Swin Transformer, DySnakeConv and Biformer respectively improves the accuracy and maintains the recall rate while keeping the mAP scores relatively stable. FPS slightly decreases with the inclusion of the Swin Transformer components, which is expected due to the increased computational load. The full configuration with all features and CIoU /SIOU loss achieves high precision and recall but does not substantially improve the mAP0.5:0.9 compared to the base model. The model combining the three architectures and EIOU achieved the best performance with a precision of 0.97, recall of 0.91, and mAPs of 0.95 (at IoU 0.5). Its computational efficiency is notable with high FPS.

Replacing CIoU with SIoU or EIoU in the full-featured model results in different trade-offs. The SIoU variant increases precision but lowers the mAP across IoU thresholds, suggesting a more conservative detection with fewer false positives but possibly more false negatives. The EIoU variant shows the best results in terms of precision and recall, and it significantly improves the mAP0.5:0.9, suggesting that this IoU loss function is better at balancing different types of detection errors across various IoU thresholds.

Overall, the inclusion of advanced architectural features and IoU loss functions seems to offer improved precision and recall, with varying impacts on the mean average precision and computational cost. The choice of configuration might depend on the specific requirements of the use case, such as the need for speed (high FPS) versus the need for accuracy (high mAP). The EIoU variant seems to offer the best overall performance at the cost of computational complexity. By evaluating the comprehensive metrics of the architecture, the YOLO-TDD algorithm for PCB defect detection was determined, incorporating Swin Transformer, DySnakeConv, Biformer, and the EIOU loss function.

Table 4 Performance Comparison of Proposed Method

model	P/%	R/%	mAP0.5/%	mAP0.5:0.9/%	FPS
Faster RCNN	0.872	0.786	0.752	0.435	133.79
SSD	0.868	0.608	0.737	0.499	90.5
Tiny-RetinaNet	0.674	0.681	0.684	0.417	110.6
Efficient	0.714	0.653	0.703	0.402	101.2
YOLOv5	0.942	0.901	0.923	0.452	256.41
YOLOv7	0.873	0.826	0.877	0.415	116.27
YOLOv8	0.851	0.864	0.892	0.462	270.27
YOLOv9	0.867	0.841	0.875	0.446	263.16
YOLOv10	0.881	0.835	0.908	0.482	**335.13**
YOLOv8-TDD	**0.979**	**0.913**	**0.957**	**0.534**	309.54

5.3 Performance Comparison of Proposed Method

Table 4 shows the performance comparison of different object detection methods and the proposed model in PCBs defect detection. Each model is evaluated based on the above common indicators:

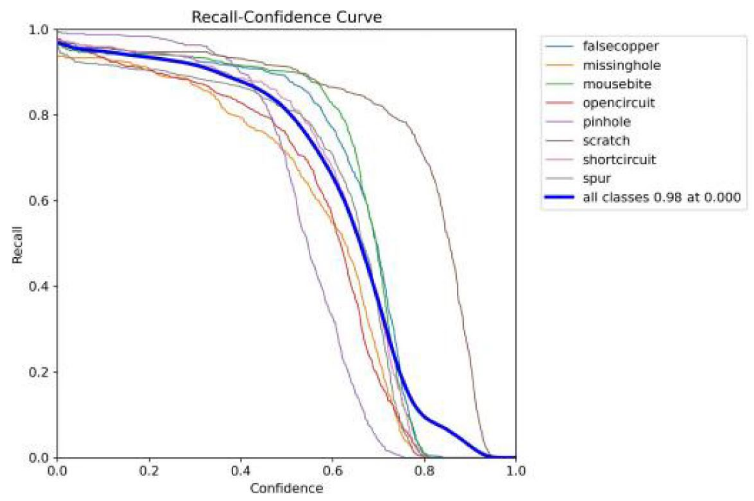
From Table 4, it is evident that YOLOv8-TDD achieves the highest accuracy and recall, and also has the highest mAP score, indicating that it is the most accurate model among all tested. On the other hand, YOLOv10 has the highest FPS.

In summary, for PCB defect detection tasks, YOLOv10 demonstrates the highest FPS, indicating that it processes more images per second, which allows for faster response in real-time applications and makes it well-suited for scenarios with high-speed requirements. In terms of accuracy, YOLOv5 and YOLOv8-TDD perform excellently, with YOLOv8-TDD striking a good balance between precision, speed, and computational efficiency. It achieves strong accuracy and recall across all categories, offering the best

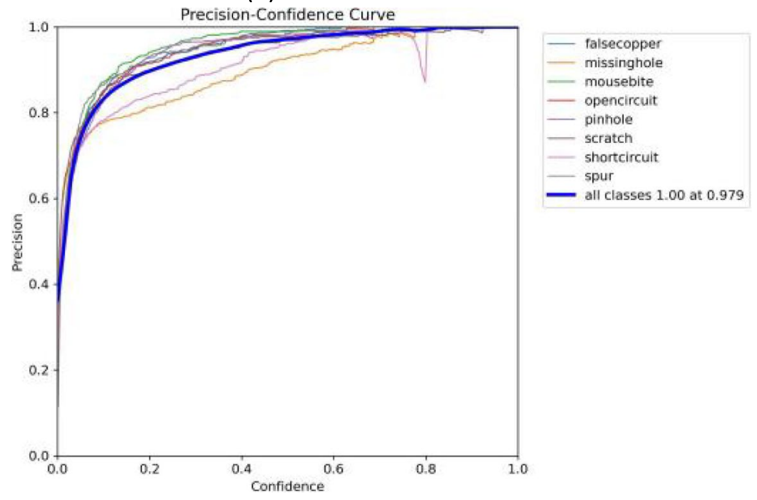
Table 3 Module Ablation Experimental Results

YOLOv8	Swin Transformer	DySnakeConv	Biformer	CIoU	SIoU	EIoU	P	R	mAP 0.5	FPS
√				√			0.85	0.86	0.89	270.2
√	√			√			0.91	0.89	0.92	222.2
√		√		√			0.87	0.87	0.90	357.1
√			√	√			0.93	0.89	0.93	333.3
√	√	√	√	√			0.91	0.89	0.93	289.3
√	√	√	√		√		0.88	0.87	0.90	312.5
√	√	√	√			√	0.97	0.91	0.95	309.5

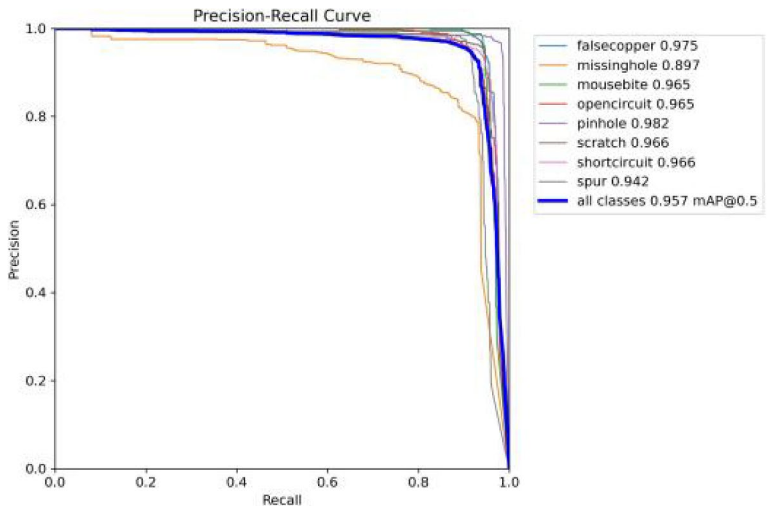
Fig.7 YOLOv8-TDD Results for PCB Defects Detection (**a**) Confidence Recall Curve (**b**) Confidence Precision Curve (**c**) Precision Recall Curve



(a) Confidence Recall Curve



(b) Confidence Precision Curve



(c) Precision Recall Curve

overall detection performance and the second highest FPS. The improved YOLOv8 stands out as the most accurate and fastest model; if larger size and increased parameter count are not limiting factors, it is a highly recommended choice.

5.4 YOLOv8-TDD Results for PCB Defects Detection

Confidence-Recall curve, (Fig. 6a) evaluates the recall of the detection system at different confidence thresholds for various defect categories. High recall at high confidence levels is desirable, indicating that the system is accurately detecting a large proportion of actual defects. From the curve, it appears that all defect categories achieve near-perfect recall (0.98) for most of the confidence interval, with a slight decrease as confidence approaches 1. This suggests that the model is very effective in detecting defects when it is confident about its predictions.

Confidence-Precision curve, (Fig. 6b) shows the precision of the defect detection system across different confidence thresholds. Precision is a measure of how many of the detected items are actual defects. The ideal scenario is high precision across all levels of confidence. The curve shows that precision for all defect categories is maintained at a high level across most confidence thresholds and reaches 1.0 at a confidence of approximately 0.979, indicating excellent precision.

Precision-Recall curve, (Fig. 6c) shows the precision of each defect category at different levels of recall. The area under this curve (AUC) gives the mean Average Precision (mAP). The provided curve demonstrates high precision across all levels of recall, which is an indicator of outstanding performance. The legend shows mAP at 0.5 IOU for all classes combined is 0.957, which is an excellent score for this type of detection task Fig. 7.

In summary, the defect detection model(YOLOv8-TDD) exhibits high performance, with excellent recall and precision across various defect categories, as evidenced by the close-to-perfect mAP of 0.957 at an IOU of 0.5. Such results indicate a well-trained model that is both accurate and reliable in detecting PCB defects, which is crucial for automating quality control in PCB manufacturing. The consistency across all categories suggests the model has been effectively generalized to various types of defects and can serve as a robust tool in industrial settings.

6 Conclusion

Our research work has successfully demonstrated the effectiveness of the YOLOv8-TDD algorithm in detecting PCB defects, achieving superior performance metrics in both detection accuracy and processing speed. Through strategic enhancements in the model architecture and training process, including dataset augmentation and optimization of the loss function, the YOLOv8-TDD algorithm shows a substantial improvement over traditional defect detection methods and existing deep learning models. These advancements substantiate the potential of deep learning-based solutions in industrial quality control, specifically in environments where precision and efficiency are paramount. Future research could explore the integration of additional deep learning innovations to further refine detection capabilities and expand the model's applicability to other complex manufacturing tasks.

Data Availability The datasets analysed during the current study are available in the Hubei University repository, https://universe.roboflow.com/hubei-university-of-technology-rmbpi/pcb-ecjga/dataset/2

Declaration

Competing Interests The authors declare no competing interests.

References

1. Zheng LJ, Zhang X, Wang CY, Wang LF, Li S, Song YX, Zhang LQ (2013) Experimental study of micro-holes position accuracy on drilling flexible printed circuit board. Innovative Solutions. https://doi.org/10.14279/depositonce-3753
2. Deng L (2019) Research on PCB surface assembly defect detection method based on machine vision. M.S. thesis, Wuhan University of Technology, Wuhan. https://link.cnki.net/doi/10.27381/d.cnki.gwlgu.2019.000586
3. Yun Z, Zhi-gang LI, Yu-qiang ZH (2020) Research progress and prospect of machine vision technology. J Graph 41(6):871
4. Khalid NK, Ibrahim Z, Abidin MSZ et al (2008) An algorithm to group defects on printed circuit board for automated visual inspection. Int J Simul Syst Sci Technol 9(2):1–10
5. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, Berg AC (2016) SSD: single shot multibox detector. In: Computer vision – ECCV 2016: 14th European conference, Amsterdam, the Netherlands, Oct. 11–14, 2016, proc, part I, vol 14. Springer, Cham, pp 21–37. https://doi.org/10.48550/arXiv.1512.02325
6. Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: unified, real-time object detection. Proc IEEE Conf Comput Vis Pattern Recognit:779–788. https://doi.org/10.1109/CVPR.2016.91
7. Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. Proc IEEE Conf Comput Vis Pattern Recognit:580–587. https://doi.org/10.1109/CVPR.2014.81
8. Girshick R (2015) Fast r-cnn. Preprint at https://arxiv.org/abs/1504.08083
9. Ren S, He K, Girshick R, Sun J (2016) Faster r-cnn Towards real-time object detection with region proposal networks. IEEE Trans Pattern Anal Mach 39(6):1137–1149
10. Wang CY, Bochkovskiy A, Liao HYM (2023) 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, pp 7464–7475. https://doi.org/10.1109/CVPR52729.2023.00721
11. Liu Z, Hu H, Lin Y, Yao Z, Xie Z, Wei Y, Ning J, Cao Y, Zhang Z, Dong L et al (2022) Swin transformer v2: Scaling up capacity

and resolution. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 12009– 12019
12. Zhang QL, Yang YB (2021) Sa-net: Shuffle attention for deep convolutional neural networks. In ICASSP 2021–2021 IEEE international conference on acoustics, speech and signal processing (ICASSP). Pp. 2235–2239. IEEE
13. Liu Z, Baida Qu (2021) Machine vision based online detection of pcb defect. Microprocess Microsyst 82:103807
14. Kim J, Ko J, Choi H, Kim H (2021) Printed circuit board defect detection using deep learning via a skip-connected convolutional autoencoder. Sensors 21(15):4968
15. Gaidhane VH, Hote YV, Singh V (2018) An efficient similarity measure approach for pcb surface defect detection. Pattern Anal Appl 21:277–289
16. Annaby MH, Fouda YM, Rushdi MA (2019) Improved normalized cross-correlation for defect detection in printed-circuit boards. IEEE Trans Semicond Manuf 32(2):199–211
17. Tsai D-M, Huang C-K (2018) Defect detection in electronic surfaces using template-based fourier image reconstruction. IEEE Trans Comp Packaging Manuf Technol 9(1):163–172
18. Cho JW, Seo YC, Jung SH, Jung HK, Kim SH (2006) A study on real-time defect detection using ultrasound excited thermography. J Korean Soc Nondestructive Testing 26(4):211–219
19. Dong JY, Lv WT, Bao XM et al (2021) Research progress of the pcb surface defect detection method based on machine vision. J Zhejiang Sci-Tech Univ 45(3):379–389
20. Ardhy F, Hariadi FI (2016) Development of SBC based machine-vision system for PCB board assembly automatic optical inspection. In: 2016 international symposium on electronics and smart devices (ISESD). IEEE, pp 386–393. https://doi.org/10.1109/ISESD.2016.7886753
21. Baygin M, Karakose M, Sarimaden A, Erhan AKIN (2017) Machine vision based defect detection approach using image processing. In: 2017 international artificial intelligence and data processing symposium (IDAP). IEEE, pp 1–5. https://doi.org/10.1109/IDAP.2017.8090292
22. Ma J (2017) Defect detection and recognition of bare PCB based on computer vision. In: 2017 36th Chinese control conference (CCC). IEEE, pp 11023–11028. https://doi.org/10.23919/ChiCC.2017.8029117
23. Deng YS, Luo AC, Dai MJ (2018) Building an automatic defect verification system using deep neural network for PCB defect classification. In: 2018 4th international conference on Frontiers of signal processing (ICFSP). IEEE, pp 145–149. https://doi.org/10.1109/ICFSP.2018.8552045
24. Huang G, Liu Z, Van Der Maaten L, Weinberger KQ (2017) Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4700–4708. https://doi.org/10.1109/CVPR.2017.243
25. Huang W, Wei P, Zhang M, Liu H (2020) Hripcb: a challenging dataset for pcb defects detection and classification. J Eng 2020(13):303–309
26. Yang Y, Kang H (2023) An enhanced detection method of pcb defect based on improved yolov7. Electronics 12(9):2120
27. Geng Z, Gong T (2021) Pcb surface defect detection based on improved faster r-cnn. Mod Comput 19:89–93
28. Ding R, Dai L, Li G, Liu H (2019) Tdd-net: a tiny defect detection network for printed circuit boards. CAAI Trans Intell Technol 4(2):110–116
29. Hu SS, Xiao Y, Wang BS, Yin JY (2021) Research on pcb defect detection based on deep learning. Electr Meas Instrum 58(03):139–145
30. Yuan J, Peng Y (2022) Defect detection method of PCB based on improved YOLOv5. Int J Front Eng Technol 4(10). https://doi.org/10.25236/IJFET.2022.041005
31. Wang SQ, Lu H, Lu D, Liu Y, Yao R (2022) Pcb board defect detection based on lightweight artificial neural network. Instrum Techniq Sens 5:98–104
32. Liu Z, Lin Y, Cao Y, Hu H, Wei Y, Zhang Z, Guo B (2021) Swin transformer: hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 10012–10022. https://doi.org/10.48550/arXiv.2103.14030
33. Qi Y, He Y, Qi X, Zhang Y, Yang G (2023) Dynamic snake convolution based on topological geometric constraints for tubular structure segmentation. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 6070–6079. https://doi.org/10.1109/ICCV51070.2023.00558
34. Zhu L, Wang X, Ke Z, Zhang W, Lau RW (2023) Biformer: vision transformer with bi-level routing attention. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 10323–10333. https://doi.org/10.1109/CVPR52729.2023.00995
35. Hubei University of Technology. Pcb dataset (2023). https://universe.roboflow.com/hubei-university-of-technology-rmbpi/pcb-ecjga. Accessed 1 Feb 2024

Gao Yunpeng obtained his Bachelor of Engineering degree from Beijing Institute of Graphic Communication in 2022. He is currently pursuing a master's degree in Mechanical and Electrical Engineering at Beijing Institute of Graphic Communication. His research focuses on defect detection based on deep learning.

Zhang Rui received the Ph.D. degree in 2008 from China Agricultural University. His research interests mainly focus on optoelectronic sensing technology, intelligent control system, and machine learning.

Yang Mingxu born in Xinzhou, Shanxi Province, is studying for Master's degree at Beijing Information Science and Technology University, mainly engaged in mechanical structure design, electromechanical transmission control, image recognition, objects de- tection and algorithm improvement.

Fahad Sabah Fahad Sabah is a senior lecturer at Faculty of CS&IT, Superior University, Pakistan. He is currently pursuing the Ph.D. degree in Computer Science with the Faculty of Information Technology, Beijing University of Technology, China. His research interests include Machine Learning, and real-world computer applications. His academic contributions are widely recognized in the international academic community, through his published research articles in reputed journals.